National College of Ireland

A Hybrid Approach for Detecting Phishing Mails Using Textual, Content, and URL Analysis with Ensemble Learning

MSc Research Project
MSc. Cybersecurity

Bejoy Asha Shajilal
Student ID: 22227067

School of Computing
National College of Ireland

Supervisor: Niall Heffernan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Bejoy Asha Shajilal		
Student ID:	22227067		
Programme:	MSc. Cybersecurity	**Year:**	2023-2024
Module:	Research Project		
Supervisor:	Niall Heffernan		
Submission Due Date:	16-09-2024		
Project Title:	A Hybrid Approach for Detecting Phishing Mails Using Textual, Content, and URL Analysis with Ensemble Learning		
Word Count:	8721 **Page Count** 30		

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Bejoy Asha Shajilal

Date: 16-09-2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A Hybrid Approach for Detecting Phishing Mails Using Textual, Content, and URL Analysis with Ensemble Learning

Bejoy Asha Shajilal
22227067

Abstract

Phishing is an important threat to the field of cybersecurity, with attackers constantly developing tactics to avoid detection systems. The study addresses the difficulty by investigating a hybrid approach to phishing detection that combines URL analysis, textual analysis, and content-based analysis with ensemble learning approaches. The major goal is to create a robust detection model that improves accuracy while minimising false positives and false negatives, hence enhancing the detection of phishing emails.

This study's data came from the Enron Corpus for legitimate emails and the Figshare-curated Nigerian dataset for phishing emails. These datasets produced key features such as BERT embeddings for textual content and numerous indications derived from HTML and URL analysis. The study used machine learning models such as Decision Tree(DT),K-Nearest Neighbour (KNN), Support Vector Machine (SVM) and Random Forest, which were then integrated with Stacking and Soft Voting ensemble approaches.

While individual models such as SVM and Random Forest performed well, ensemble techniques demonstrated better balanced performance across evaluation metrics. The Stacking Ensemble, in particular, displayed the capacity to combine the strengths of textual and content-based features, earning 96.18% accuracy and an F1-Score of 0.7846.

The findings indicate that, while a hybrid method is helpful, it still requires additional development, particularly in terms of improving ensemble techniques to better capture the complex nature of phishing emails. This study adds to the ongoing development of improved phishing detection systems and establishes the framework for future research into improving real-time email filtering systems in dynamic cybersecurity contexts.

1 Introduction

Despite increased attention to the problem of phishing, it remains a major threat to computer security, with constant shifts in tactics by attackers staying one step ahead of detection systems. For example, in 2022, more than 3.4 billion attempted phishing emails were sent every day, putting phishing among the most prevalent forms of cybercrime. May 2022 is a

landmark date in this respect because during this single month the total number of unique phishing webpages, recognized by anti-phishing organizations like APWG, increased by 61%. Moreover, nearly 84% of phishing sites now feature SSL certificates of their own, making detection still more difficult since it is no longer easy just to identify insecure pages. This dramatic rise calls for an urgent upgrade in detection skills if the users and companies trying to defend themselves are not to be overwhelmed by these concerted attacks.

Modern phishing tactics, such as the use of SSL certificates, impersonation of established brands, and sophisticated URL redirection techniques, have grown in complexity. As of 2023, more than 45% of phishing indicators involved misusage of legitimate URLs for redirection, planting malicious QR codes in them; conventional security measures overlook these altogether. Also in 2023, Proofpoint's annual security report found that 84% of organizations questioned had suffered at least one successful phishing attack last year and that there had been 76% more financial losses than in 2021. These data confirm the need for a many-techniques approach to combating phishing that draws together a variety of research angles in order to grasp the full panoply of strategies E-criminals so determinedly employ.

This thesis attempts to answer the following question: “How can the combination of URL analysis, text analysis and content analysis using ensemble learning improve the detection of phishing mail?” This study has a number of objectives:

- To evaluate how well URL analysis, text analysis, and content-based analysis alone detect phishing emails.
- Create a hybrid model combining these techniques by using ensemble learning.
- Assess the validity of this model against individual detection systems currently in use.
- Determine how well the model can be employed in the real world for email filtering systems, in view of phishing accusers constantly changing tactics.

This study is aware of various limitations, such as its reliance on both the quality and diversity of the dataset that trained the detection model. Moreover, the assumption is made that those features selected constituting each type of analysis suffice to make clear the subtleties of phishing attempts. These limitations will be further discussed in the methodology and results sections.

This paper is constructed as follows: The Literature Review chapter will discuss the shortcomings of current anti-phishing technology, preparing the ground for alternative methods to pursue. The Methodology chapter will go into detail on the design of a hybrid model developed in this study, its construction procedure – setup, feature selection and training. The Results section will present what has been achieved by our experiments. Having done that, a Discussion will set these findings into the wider context of research in phishing

detection. Finally, the Conclusion will sum up our achievements and suggest possible future lines of inquiry for researchers in this area.

2 Related Work

Phishing poses a challenge, in the cybersecurity landscape leading to the advancement of detection methods. With phishing attacks growing sophisticated traditional detection strategies like blacklists and signature-based systems are proving effective. This has prompted a shift towards techniques, particularly machine learning (ML) and deep learning (DL) methods. These approaches provide adaptive solutions that can keep up with the changing tactics employed by cybercriminals. The review explores the progress and application of these techniques highlighting the insights, from recent studies in the realm of phishing detection.

2.1 Traditional Detection Methods

Historically, phishing detection depended on methods like blacklists, which block known malicious URLs, and signature-based techniques that identify specific patterns within emails or web pages. Even though these techniques are effective in certain cases they had limitations due to the static nature. For instance, blacklists needs to be constantly updated inorder to stay relevant, and signature-based methods can be easily bypassed by novel phishing techniques that have yet to be cataloged (Huang et al., 2019). The traditional detection methods failed to detect new or modified attacks thus leading to a higher false negative rate

Nosseir, Nagati and Taj-Eddin, (2013) pointed out the limitations of methods by suggesting a spam filter based on character word patterns using a neural network classifier. They showed how applying ASCII value based weight normalization could enhance detection accuracy. However the model faced challenges with a high False Positive Rate (FPR) meaning that while it could catch many phishing attempts it also mistakenly identified legitimate emails as phishing. This early research paved the way, for studies that stressed the importance of sophisticated detection methods to decrease false alarms without compromising sensitivity.

Gupta and Manickam, (2013) introduced the Phishing Dynamic Evolving Neural Fuzzy (PDENF) framework, which blended supervised and unsupervised learning techniques to identify zero-day phishing emails. Their model stood out for its flexibility enabling it to learn from data in real time and adjust its settings accordingly. This ability to adapt and enhance over time marked an advancement, in phishing detection. The frameworks focus on negative rates and overall accuracy mirrored a growing awareness that effective phishing detection demands models that can differentiate increasingly subtle differences, between legitimate and malicious content.

Hamid, Abawajy and Kim, (2013) took a unique approach by focusing on the behavior of email senders rather than the content of the emails themselves. Their model used a Naïve Bayes classifier to study sender behavior patterns, achieving a high accuracy rate of 94%.

This approach that focused on behavior proved effective in detecting phishing attempts that could bypass content based filters. The study also emphasized the importance of further research into profiling attacker behaviors, suggesting that combining behavioral analysis with content based techniques could improve overall detection accuracy.

In another study by GhaziM.Jameel and E. George, (2013), they explored the use of neural networks for detecting phishing attacks. They employed a feed forward neural network to categorize emails based on features extracted from HTML headers and bodies. Their model achieved an impressive accuracy rate of 98.72% with minimal processing times, showcasing the efficiency of neural networks in detecting phishing attempts. This research highlighted how neural networks have great potential to quickly analyze large amounts of data, making them suitable for real time detection applications.

2.2 Machine Learning Approaches

The limitations of traditional methods have lead to a big interest in machine learning (ML) as an alternative flexible and adjustable approach towards identification of phishing attempts. With ML, the models can examine data and find signals, patterns or variations that may be indicative of phishing without being constructed around rules (static) or signatures. Since phishing tactics are always changing, the ML model offers special effectiveness in such an environment.

Korkmaz, Sahingoz and Diri, (2020)proposed a model that focused on the analysis of URLs, considering features such as URL length, presence/non-presence of special characters and occurrence of specific keywords to classify phishing sites. Their work demonstrated good accuracy indicating the importance of thoughtful feature engineering in ML driven phishing detection. The model was able to identify multiple phishing techniques by looking out for discriminating features in URLs which were not caught by most of the rule-based traditional systems, confined with strict rules. This finding highlights the importance of using ML techniques to make anti phishing systems more robust.

Adebowale, Lwin and Hossain, (2020)integrated image, frame and textual features into a web oriented phishing detection & defence system making the application of ML even larger. Machine learning classifiers along with a number of web content features helped in increasing the detection accuracy to a greater precision. The above method combines various data sources and reminds us of the importance of improving detection accuracy. This model combines visual, structural and textual data to help detect malicious phishing that might be missed when only using one type of data.

The Optimal Feature Selection Neural Network (OFS NN) model developed by Zhu *et al.*, (2023) was another significant addition. This proposed approach to detects phishing Web page using optimal feature selection techniques and neural networks. OFS NN model confirmed that by carefully selecting input features we can achieve high detection accuracy and at the same time decrease computational costs. OFS NN is suited for real time applications due to its fast detection process with high accuracy.

2.3 The Impact of Deep Learning Models

Deep learning, a branch of machine learning, has had a significant impact on detecting phishing scams by effectively recognizing intricate patterns in large datasets without the need for manual feature engineering. Unlike traditional machine learning approaches, deep learning models can autonomously identify and understand key features from raw data, making them highly efficient for analyzing unstructured information like emails and web content.

In a groundbreaking study by Fang *et al.*, (2019) an innovative phishing detection system named THEMIS was introduced, using Recurrent Convolutional Neural Networks (RCNN). By using attention mechanisms and multi-level vectors to assess various components of email structures such as words, characters, headers and bodies at different levels of detail, THEMIS achieved an impressive accuracy rate of 99.85%. This highlights the effectiveness of deep learning in detecting even subtle phishing schemes based on the hierarchical organization of phishing emails.

(Harikrishnan *et al.*, 2019)delved into the application of deep learning techniques like DNN (Deep Neural Networks), RNN (Recurrent Neural Networks) and CNN (Convolutional Neural Networks) in classifying malicious URLs. Their research highlighted that employing time split pre-processing alongside decision tree classifiers and term frequency inverse document frequency (tf idf) representation produced optimal outcomes with an accuracy level reaching 88.5%. This research study emphasized the benefits of integrating various machine learning methods for identifying phishing attempts, while also acknowledging the challenges associated with limited data sets. The researchers stressed the importance of having larger and more diverse data sets to enhance the applicability of advanced deep learning models.

In a study conducted by Ali and Ahmed, (2019), they adopted a hybrid strategy by creating an intelligent model for predicting phishing websites. This model combined deep neural networks with a genetic algorithm to select and weigh features effectively. Their model achieved an accuracy rate of 95%, showcasing how hybrid models can enhance phishing detection capabilities. By utilizing a genetic algorithm, the model could dynamically optimize feature selection, improving its ability to identify various phishing strategies.

Wei *et al.*, (2019)introduced a convolutional neural network (CNN) based model tailored specifically for detecting URL based phishing attacks. Their approach involved treating URLs as character sequences and leveraging convolutional layers to recognize unique characteristics in phishing URLs. This method proved highly successful, particularly when applied to extensive data sets, underscoring the scalability and efficiency of CNNs in combating phishing threats.

2.4 Hybrid Models and Ensemble learning

Researchers have been motivated to investigate hybrid models that merge various analytical methods due to the limitations of single method detection systems. These models harness the strengths of different detection approaches, leading to more resilient and precise systems.

In their work, Hota, Shrivas and Hota, (2018) crafted an ensemble model that combines diverse machine learning classifiers to enhance phishing detection accuracy. Their model

employs a feature selection method to remove irrelevant features, thereby refining the detection process. By adopting an ensemble strategy, the model can leverage the advantages of different classifiers while addressing their individual shortcomings. The study illustrated that ensemble learning can notably enhance detection accuracy, especially in intricate phishing scenarios where single method approaches may falter.

Similarly, Janjua et al. (2020) delved into hybrid models in their exploration of supervised machine learning techniques for managing insider threats like phishing. Their model integrates multiple classifiers to boost detection accuracy, particularly in situations where phishing emails closely mimic legitimate communications. This research emphasized the significance of ensemble techniques in striking a balance between sensitivity and specificity, ensuring that the model can effectively identify phishing attempts without triggering excessive false positives.

Furthermore, Ding *et al.*, (2019)introduced a keyword based fusion approach for identifying phishing webpages. Their study incorporated elements like keyword frequency, URL format and HTML content, showing that merging diverse data types can encompass a wider array of phishing strategies. By taking this approach, they were able to decrease the occurrence of false negatives, underscoring the efficacy of hybrid models in spotting phishing attempts.

In their work, Muralidharan and Nissim, (2023) devised a sophisticated ensemble learning framework that eliminated the necessity for manual feature crafting. Their model achieved an impressive Area Under the Curve (AUC) value of 0.993, demonstrating its capability in identifying phishing attacks across different scenarios. A standout feature of their methodology was its capacity to maintain high precision while upholding user privacy—a crucial aspect in today's data centric environment. This study underscores the ongoing trend towards automated and privacy conscious phishing detection methods that can adapt to the changing landscape of cyber threats.

Bountakas and Xenakis, (2023) introduced the HELPHED (Hybrid Ensemble Learning Phishing Email Detection) framework as a solution that combines ensemble learning with hybrid features to address imbalanced datasets challenges. Their framework incorporates a range of machine learning techniques and feature sets—including content based, structural and behavioral attributes—to enhance the accuracy of identifying phishing emails. The HELPHED framework achieved an F1 score of 0.9942, even when dealing with datasets that have a significant imbalance between legitimate and phishing emails. This shows how effective and reliable the framework is in real world situations. The study emphasizes the importance of using hybrid and ensemble methods to overcome the limitations of traditional phishing detection techniques, especially in environments where phishing emails are crafted to appear very similar to genuine communications.

One key feature of the HELPHED framework is its capability to address imbalanced datasets, which is a common challenge in detecting phishing attacks due to the overwhelming number of legitimate emails compared to phishing ones. By combining various detection methods, HELPHED strengthens the system's ability to identify phishing emails while managing the large volume of legitimate messages effectively. This not only enhances detection accuracy but also reduces false positives, preventing disruptions in business operations caused by misclassifying genuine emails as potential phishing threats.

2.5 Limitations and Future Directions

The progress that we have made in the detection of phishing attempts, has a few more hurdles to jump over. The core problem is that current models are not able to generalize properly across languages and phishing attack templates. Nearly all the state-of-the-art models are trained on datasets with predominantly English phishing emails, making them potentially ineffective at identifying other languages. The other problem at hand is that cyber attackers are always changing their tactics, using practices like legitimate domains or modifying the URL on a regular basis to bypass detection. Ultimately, these emerging tactics emphasize the need for models to be flexible enough in order respond rapidly to new forms of attack.

One of the biggest challenges is that deep learning algorithms often require enormous datasets. Large amounts of data are often needed to train deep learning models, but the process of collecting labeled or unlabeled samples is very expensive in terms of human resources. In addition, models trained on a specific datasets may not be effective with new or unseen data if the phishing techniques used are very different from those in training data. The challenge, then, arises from the need for models that learn well with limited data — or can adapt to new information real-time without demanding significant retraining.

Given the evolution of phishing techniques, future work should focus on developing models that can adjust to emerging attack strategies specifically involving some social engineering or more sophisticated obfuscation. Furthermore, the input of explanations and interpretability into deep learning models is essential for deployment across a practical use-case.

Also, the combination of different detection techniques into a unified model can be defined as one path for phishing scam detection in future. Hybrid and ensemble models such as HELPHED have shown that combining different methods can provide better protection against phishing threats. More research needs to be done in the direction of making not just these basic frameworks combining ML and DL but also approaches integrating insights from fields like behavioral psychology or network analysis resulting in sophisticated, adaptable phishing detection systems.

3 Research Methodology

The research procedure for this study was designed to systematically explore the effectiveness of a hybrid phishing detection system that integrates textual analysis with content-based analysis. The study was conducted in several phases, each of which is detailed below.

3.1 Data Collection

The first phase of the research involved the collection of datasets that would provide a robust foundation for model training and evaluation. Two datasets were selected based on their relevancy and comprehensive nature:

- **Enron Corpus:** This dataset was chosen as the source of legitimate emails. The Enron Corpus is one of the most widely used datasets in email-related research, consisting of emails from employees of the Enron Corporation. This dataset offers a diverse range of email formats, topics, and structures, making it ideal for training models to recognize legitimate email communications.
- **Figshare Curated Nigerian Dataset:** This dataset was selected for its focus on phishing emails, particularly those related to Nigerian scams. It provides a realistic set of phishing attempts that challenge the models' ability to detect malicious intent. This dataset includes various types of phishing tactics, such as fraudulent financial requests, fake lottery winnings, and impersonation of legitimate entities.

3.2 Data Preprocessing

After the data was collected, it underwent a rigorous preprocessing phase to ensure consistency and quality. Preprocessing was carried out using custom Python scripts, with the "clean copy.py" script playing a pivotal role in this phase.

- **Parsing and Cleaning:** The script parsed the raw email data, extracting relevant fields such as the subject line, body text, and headers. It also removed extraneous information that could introduce noise into the analysis, such as irrelevant metadata.
- **Standardization:** The preprocessing script standardized key elements of the emails, including email addresses and subject lines. This standardization was crucial for ensuring that the data was consistent across the entire dataset, allowing for more reliable feature extraction and model training.
- **Handling Imbalanced Data:** The phishing dataset was significantly smaller than the legitimate email dataset, leading to an imbalance. To address this, stratified sampling was used during the data split to maintain the class distribution in both training and test sets.

3.3 Feature Extraction

The next phase involved feature extraction, which was conducted in two parallel streams: textual features and content-based features.

Textual Features (BERT): The textual content of the emails, primarily the body text, was processed using BERT (Bidirectional Encoder Representations from Transformers).BERT was selected because it can extract the semantic meaning and context from text, which is crucial for detecting the complex language cues that are frequently found in phishing emails. The process involved tokenizing the text and generating numerical embeddings that represent the email content. These embeddings were then used as input features for the machine learning models.

Content-Based Features: In parallel with the textual analysis, content-based features were extracted using custom scripts. These features included HTML code, the presence of forms and scripts, hyperlinks, and specific keywords associated with phishing attempts. The

features were encoded and standardized to ensure compatibility with the machine-learning models.

3.4 Model Development

The extracted features were used to train several machine-learning models. The development and training phase was divided into two parts: textual analysis and content-based analysis.

In textual analysis, BERT embeddings were fed into four machine learning models: K-Nearest Neighbors (KNN), Decision Tree, Random Forest, and Support Vector Machine (SVM). Each model was trained using Grid Search with 5-fold stratified cross-validation. This approach involved systematically exploring a predefined set of hyperparameters to identify the configuration that resulted in the best performance for each model. The models were evaluated on their ability to accurately classify phishing and legitimate emails based on the textual content.

In Content-based analysis, the content-based features were used to train the same set of models. The focus was on identifying patterns and anomalies in the structural elements of the emails, such as the presence of suspicious scripts or forms. The models were trained and evaluated using the same Grid Search and cross-validation approach as in the textual analysis.

Using ensemble learning techniques, the best-performing models from both analyses were combined to further improve the detection accuracy. Stacking Ensemble and Soft Voting were the two ensemble techniques that were used. The meta-learner in the Stacking Ensemble method was a Multi-Layer Perceptron (MLP), which was trained to combine the outputs of the base models (SVM and Random Forest). However, to arrive at the final decision, the Soft Voting method averaged the probabilities predicted by the base models.

3.5 Evaluation Methodology

The methodology involved the selection of appropriate metrics, the application of statistical techniques, and the rationale behind these choices.

3.5.1 Evaluation Metrics

A comprehensive set of metrics was selected to evaluate the performance of the models, ensuring that the evaluation was both detailed and reliable:

F1-Score: The F1-Score was selected as the main evaluation metric because it can balance recall and precision, which makes it especially useful for datasets that are unbalanced and where misclassified legitimate emails or false negatives (phishing emails missed) can have serious consequences.

Accuracy: Although widely used, accuracy alone can be misleading in imbalanced datasets. As a result, even though it was computed, another metric was also used to assess the performance of the model.

Precision and Recall: Recall measures the percentage of real phishing emails that were correctly identified, whereas precision measures the percentage of correctly identified phishing emails out of all emails classified as phishing. These metrics collectively provide insight on the trade-off between obtaining all phishing emails and minimizing false positives.

Area Under the Curve (AUC): The AUC metric was employed to evaluate the model's overall performance in distinguishing between phishing and real emails at various threshold settings.

Matthews Correlation Coefficient (MCC): The MCC was included as it provides a balanced measure of the model's performance, considering all four outcomes of the confusion matrix (true positives, false positives, true negatives, and false negatives). MCC provides a more detailed picture of model performance and is especially helpful in situations where datasets are unbalanced.

3.5.2 Statistical Techniques and Analysis

The statistical analysis of the models' performance was carried out using several techniques to ensure robustness and reliability:

- **5-Fold Stratified Cross-Validation:** This technique was employed during model training to assess the models' ability to generalize to new, unseen data. The dataset was divided into five folds and ensured that each fold had a similar distribution of phishing and legitimate emails which aimed to produce more reliable and generalizable models.
- **Grid Search for Hyperparameter Tuning:** For each model, several hyperparameter combinations were methodically investigated using Grid Search. With the use of this technique, it was possible to determine which configuration was best for each model, guaranteeing peak performance during the evaluation stage.
- **Confusion Matrix Analysis:** The confusion matrix was used to provide a detailed breakdown of the models' predictions, including true positives, false positives, true negatives, and false negatives. This analysis was critical for understanding the specific strengths and weaknesses of each model, particularly in terms of its ability to correctly identify phishing emails without misclassifying legitimate ones.

The choice of evaluation metrics and statistical techniques was based on a careful consideration of the challenges posed by phishing detection. The F1-Score, AUC, and MCC were selected as the main metrics for assessment due to the dataset's imbalance and the significance of reducing false positives and false negatives. The use of stratified cross-validation and Grid Search ensured that the models were rigorously tested and optimized, providing confidence in the reliability of the results.

4 Design Specification

The proposed phishing detection system is designed to incorporate the best features of both text and content-based techniques, giving it an advantage in predicting with accuracy while eliminating as many false positives as feasible. It is built on a modular architecture that allows any component to work independently of the rest of the system while also contributing to overall performance and growth.

There were a number of primary goals in creating this phishing detection system. This most important need is to have great detection accuracy and less false positives as possible. This was especially important as phishing emails nowadays are crafted to look like an official message even that may have been a challenge. In order to achieve this goal, the system was designed as a hybrid text and content-based analysis tool, in such way that both techniques are helping each other giving a more comprehensive view on emails being analysed.

The other important piece was how flexible and extendable system should be. Since BERT processes text for analysis, the system will be able to do a thorough job of parsing any kind of email content — even more complex and sophisticated types. The modular design of the system also supports upgrades and adaptations, meaning that new features or models can be added later if required.

Finally, the system was intended to be efficient. We employed the Stacking Ensemble with an MLP meta-learner, which was intended to strike a balance between accuracy and computing complexity. Taking the average of multiple model outputs, contributes to a system with high results that can operate in real time without heavy processing requirements.

5 Implementation

The implementation phase of this research project was a critical step where theoretical plans and methodologies were translated into practical actions. This phase involved several stages, each designed to systematically process data, train machine learning models, and evaluate their performance. The ultimate goal was to develop a robust phishing detection system capable of accurately distinguishing between legitimate and phishing emails. Below is a detailed account of the implementation process, including data preprocessing, feature extraction, model development, ensemble learning, and the outputs generated.

5.1 Data Preprocessing

Data processing was the first step used in our implementation and it is a very important process to make sure that the raw data is converted into a single, clean, consistent format. These datasets include a portion of the Enron Corpus as well as an intentionally curated

Nigerian dataset curated by Figshare. The Datasets were very noisy and unorganized that required a good amount of pre processing before they could be fed to the models.

5.1.1 Data Cleaning:

Custom Python scripts were written to automate the data cleaning process. The primary focus was on eliminating with noise data (like repeated html tags, any uninformative metadata and various special characters which were not really adding to meaning of content). Furthermore, the scripts convert email addresses to lower case and trim any extra spaces or special characters in order to keep data homogenous.

5.1.2 Data Transformation:

Once cleaned, the next step was to convert this text data into a format suitable for machine learning use. We used BERT (Bidirectional Encoder Representations from Transformers) embeddings for the email body text. These are just high-dimensional vector representations of word meanings that allow the models to understand in what context words show up. The text was tokenized using BERT tokenizer which not only not just does a great job at splitting apart the words into tokens but also took care of preserving the semantic relationship between tokens. These tokens are then passed through the BERT model to generate embeddings, which were aggregated to a single vector for each email. This output was then stored in NumPy arrays and then converted to a data frame, which can be used as features to train the model

Additionally, we wrote custom scripts to detect content-based features such as the presence of HTML forms, scripts, and hyperlinks etc. These features were parsed using the tools like Regex for pattern matching and BeautifulSoup to parse elements in HTML. Categorical features were then encoded using one-hot encoding, this is in order to convert them into a binary format such that they can be worked with machine learning models. The numeric features with high variance (i.e., for word count, number of hyperlinks etc), were standardize so that they have a mean of zero and a standard deviation of to reduce the influence of outliers on result and to ensure that all features are given equal importance in training phase.

5.1.3 Data Splitting:

The data were split into training, test, and validation datasets using stratified sampling due to imbalance between the class labels. Thus models could be trained and assessed on representative samples while ensuring that the class distribution remains constant throughout all datasets, reducing the risk of bias. 60% of the data is assigned to training set and remaining 40% is equally divided between test and validation sets. The training data is used

to train all ML models and ensemble models. Testing data is used to test the ML models in grid search and cross validation processes in text and content analysis. Whereas validation data set is used as a test set for ensemble models alone to prevent any data leakage.

5.2 Feature Extraction

Feature extraction was a critical aspect of the implementation, as it involved selecting and preparing the most relevant features from the emails that would be used to train the machine learning models. The features were categorized into two main types: textual features and content-based features.

5.2.1 Textual Feature Extraction:

The primary textual feature used in this study was the BERT embeddings generated from the email body. BERT's ability to capture the nuanced meaning of words in their context made it an ideal choice for analyzing the content of phishing emails, which often use deceptive language. These embeddings were stored as feature vectors and provided a rich, context-aware representation of the email text.

5.2.2 Content-Based Feature Extraction:

In addition to the textual analysis, several content-based features were extracted to capture the structural and contextual elements of the emails. These included:

HTML Tags and Forms: The presence of HTML tags and forms, which are often used in phishing emails to capture sensitive information, was detected and quantified. Using BeautifulSoup, the scripts identified and counted these elements, providing a binary feature that indicated their presence or absence.

Scripts and Embedded Code: Phishing emails often contain scripts or embedded code designed to execute malicious actions, such as redirecting users to fraudulent websites. Custom scripts were used to detect these elements, encoding their presence as binary features.

Hyperlinks: The number of hyperlinks in the email, as well as the use of IP addresses or non-standard ports in URLs, were extracted using Regex. These features were essential for identifying phishing attempts that relied on misleading or dangerous links.

Keyword Frequency: Certain keywords, such as "urgent" or "password," are commonly used in phishing emails to create a sense of urgency. The frequency of these keywords was calculated and included as a feature in the model.

The extracted features were combined into a comprehensive dataset, with each email represented by a feature vector that included both textual and content-based information. This dataset was then used for model training.

The following figure shows the features utilised in our study:

Feature Name	Description
Content	Body of email
contain_urls	Indicates if the email contains URLs (binary feature)
no_of_urls	Number of URLs present in the email
HTML_Code	Presence of HTML code in the email's body
HTML_Forms	Presence of HTML forms in the email's body
Scripts	Presence of scripting code in the email's body
Image_Link	Presence of a hidden hyperlink behind an image in the email's body
Bad_words_body	Number of bad words that appear in the email's body text
Bad_words_subject	Number of bad words that appear in the email's subject
Absence_of_RE	Presence of "RE" in the email's subject
Num_words_body	Number of words in the email's body text
Num_characters_body	Number of characters in the email's body text
Richness	The ratio of the total number of words to the total number of characters
Num_distinct_words_body	Number of distinct words in the email's body text
Num_email_parts	Number of different parts or sections in the email
Num_email_recipients	Number of recipients in the email
IP_URL	Presence of IP addresses instead of domain names in URLs
Num_hyperlinks	Number of hyperlinks in the email
Text_hyperlink	Presence of human-readable text that hides a hyperlink
Num_different_HREF	Number of different HREF attributes in the email
Num_dots	Maximum number of dots in URLs within the email
Port_URL	Presence of port numbers in URLs within the email
Check_domain	Presence of a different sender's domain compared to the domain in hyperlinks
Label	Target variable indicating the classification (e.g., spam or not spam)

Figure 1: List of Features Used

5.3 Model Development and Training

The model development and training phase involved applying machine learning algorithms to the extracted features to create predictive models capable of detecting phishing emails. This phase was iterative, with several models being trained, tested, and refined to achieve the best performance.

5.3.1 Model Selection:

Several machine learning models were chosen for their suitability to the classification task at hand. These models included K-Nearest Neighbors (KNN), Decision Tree, Random Forest, and Support Vector Machine (SVM). Each of these models has distinct strengths:

K-Nearest Neighbors (KNN): Known for its simplicity and effectiveness in handling multi-dimensional data, KNN was selected for its ability to classify emails based on their proximity to known examples in the feature space.

Decision Tree: This model was chosen for its interpretability and ability to capture non-linear relationships between features. It works by constructing a tree-like model of decisions, where each node represents a feature and each branch represents a decision rule.

Random Forest: As an ensemble of decision trees, Random Forest was utilized for its robustness and ability to reduce overfitting. It works by averaging the results of multiple decision trees, improving overall classification accuracy.

Support Vector Machine (SVM): SVM was applied for its strength in high-dimensional spaces and its effectiveness in handling imbalanced datasets. It works by finding the hyperplane that best separates the classes in the feature space.

5.3.2 Training Process:

To optimize the performance of these models, a Grid Search was conducted in conjunction with 5-fold stratified cross-validation. Grid Search systematically tested different combinations of hyperparameters for each model, including KNN,DT,RF and SVM, to identify the best parameters. For text analysis the grid of parameters were defined as: KNN with no: of neighbours [3, 5, 7, 10]; DT with maximum depths [None, 10, 20, 30] and split criteria ['gini', 'entropy']; RF with numbers of trees [50, 100, 200], maximum depths [None, 10, 20, 30], and split criteria ['gini', 'entropy']; and SVM with regularization parameters [0.1, 1, 10] and kernels ['linear', 'rbf', 'poly', 'sigmoid']. A similar grid was used for content analysis as well. Cross-validation was used to train and assess the models in order toget reliable estimates. The models with the highest F1-Scores were chosen after Grid Search, with Random Forest outperforming SVM in content-based analysis and SVM scoring highest in text analysis.

5.3.3 Model Evaluation:

The trained models were evaluated on a set of performance metrics, including F1-Score, Accuracy, Precision, Recall, Area Under the Curve (AUC), and Matthews Correlation Coefficient (MCC). The F1-Score was selected as the primary metric due to its ability to balance precision and recall, which is crucial in the context of phishing detection where both false positives and false negatives can have significant consequences.

After evaluation, the best-performing models from both the textual and content-based analyses were selected for further refinement and integration into the ensemble learning methods.

5.4 Ensemble Learning

To improve the overall efficacy of phishing detection system and its robustness, ensemble learning techniques were deployed. The process of training multiple models to harness the strengths and mitigate the weaknesses of each is called ensemble learning.

5.4.1 Stacking Ensemble

In this method, Stacking Ensemble was also used to combine the top performing models from previous phase. We concatenated the predictions of SVM (the best model among textual models) and Random Forest (best content-based approach) using Multi-Layer Perceptron (MLP) as a meta-learner. This MLP, which was implemented using TensorFlow, is learned to combine the outputs, which are then classified through the final classifier. By this method, the system was able to capitalize on each individual model's unique strengths. MLP Classifier has hidden layers of sizes (150, 100) and early stopping to prevent overfitting. Accuracy was assessed on the validation set, and the stacking model was trained using a combination of features from content-based and text analyses.

5.4.2 Soft Voting

Apart from Stacking, a Soft Voting approach was used here — the probabilities predicted by base models were averaged to make final decision. This method is more straightforward than Stacking but effective, especially if we have strong individual base models. Soft Voting, reduces the inconsistency in different models; in other words Soft voting is very useful way for more stable and reliable predictions.

Both ensemble methods were trained and evaluated for improving performance of models.

6 Evaluation

This evaluation helps to reveal how well the phishing detection system proposed in this study has performed; its strengths and weaknesses, long-term consequences with both academic usecases and real-world problems. This section shows a detailed analysis of the experimental results focusing on accuracy, presicison, recall, F1-Score and relevant metrics etc. The evaluation also considers the implications of these findings, for further research and practical deployments.

6.1 Model Performance Overview

The study tested four machine learning models—K-Nearest Neighbors (KNN), Decision Tree, Random Forest, and Support Vector Machine (SVM)—across both textual and content-based features extracted from emails. Additionally, two ensemble models were developed: a Stacking Ensemble with a Multi-Layer Perceptron (MLP) as the meta-learner, and a Soft Voting Ensemble.

6.1.1 Accuracy and F1-Score

The accuracy and F1-Score of each model were key indicators of their performance.

In text based analysis, SVM had highest F1 score of 0.9986 and an accuracy of 99.86% and in content based analysis, RF had the highest F1 score of 0.9832 and an accuracy of 98.36% thus outperforming the other models. And these were selected as base learners for the ensemble model.

6.1.2 Ensemble Models

The ensemble models were designed to leverage the strengths of the individual models. The Stacking Ensemble, which combined the best-performing textual and content-based models (SVM and RF), achieved an accuracy of 96.18% and an F1-Score of 0.7846. The Soft Voting Ensemble also performed admirably, with an accuracy of 95.18% and an F1-Score of 0.6845.

6.1.3 Confusion Matrices

The confusion matrices for each model provided insights into the distribution of true positives, true negatives, false positives, and false negatives. The SVM model, for example, correctly identified 99% of phishing emails (true positives) and correctly classified 99.95% of legitimate emails (true negatives) in text analysis and in content analysis RF model identified a true positive percentage of 86.5 and true negative percentage of 99% .The following figure displays the confusion matrices of text based analysis of all 4 classifiers.

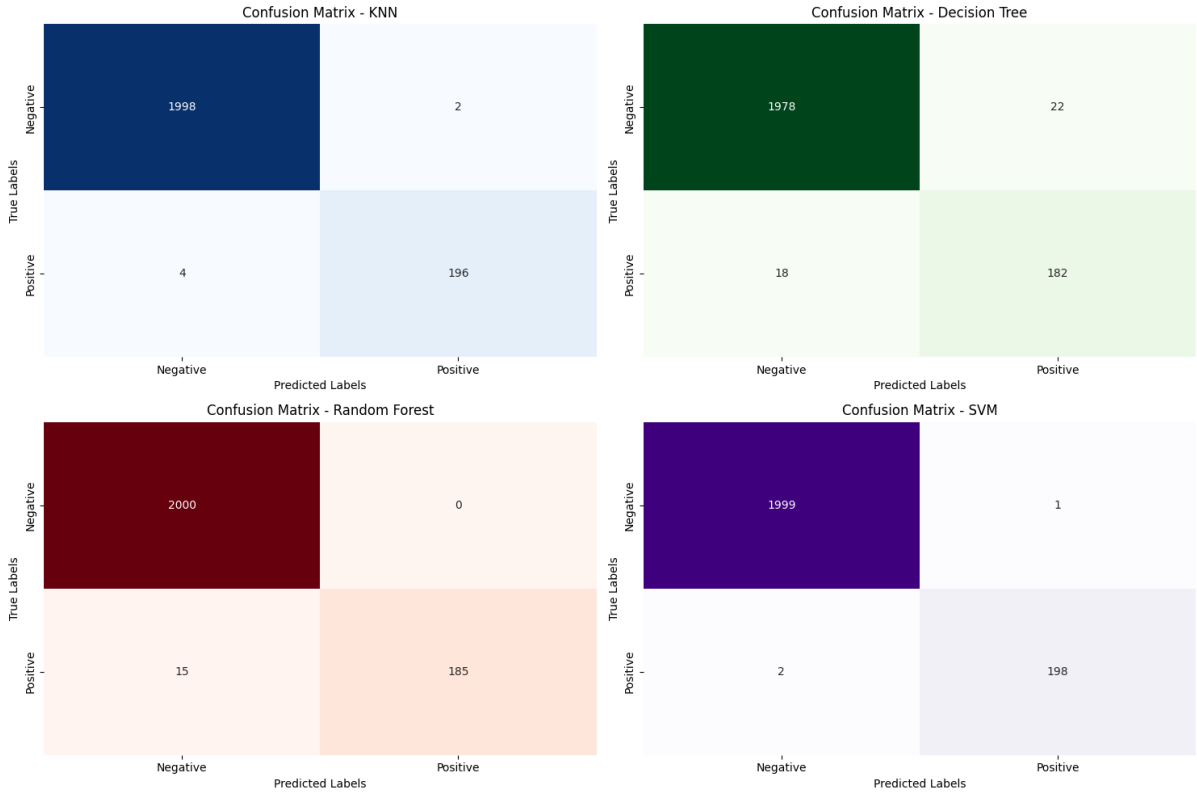

Figure 2: Confusion Matrix - Textual Analysis

Figure below shows confusion matrix of content based analysis of all 4 classifiers:

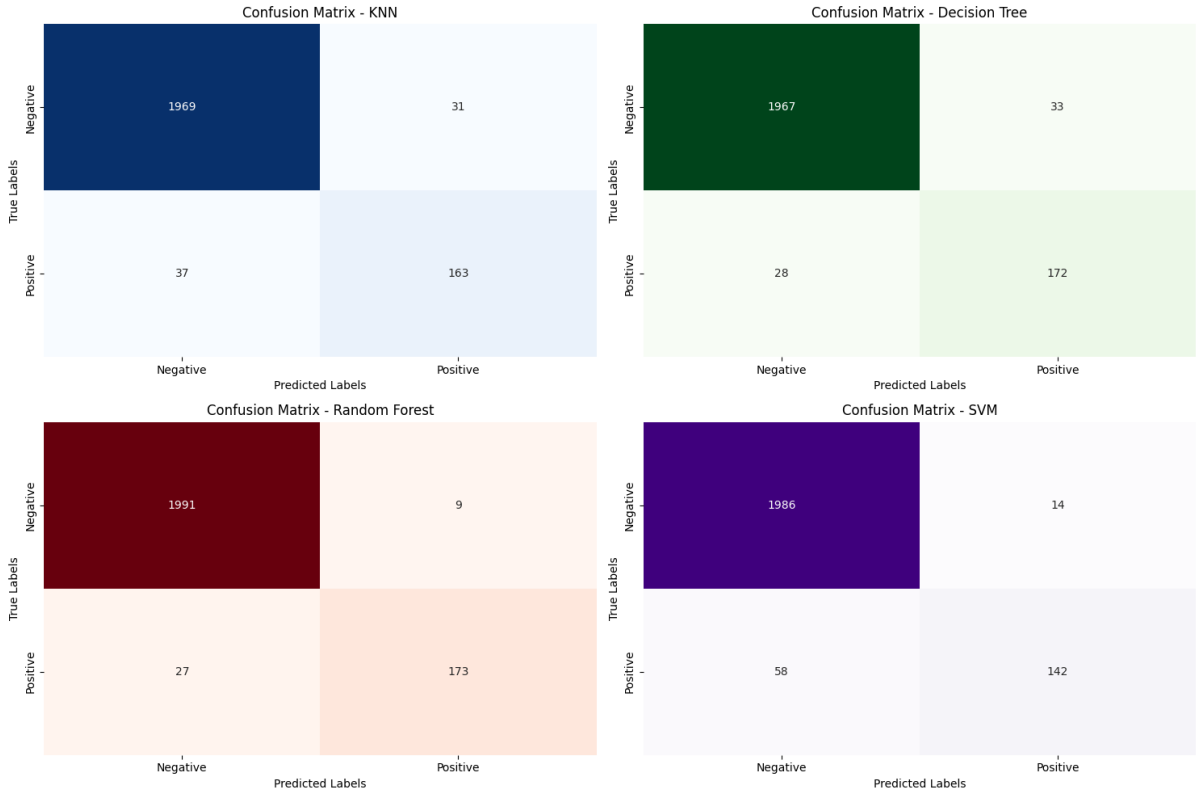

Figure 3: Confusion Matrix - Content analysis

Figure below shows the confusion matrix for stacking and soft-voting ensemble models:

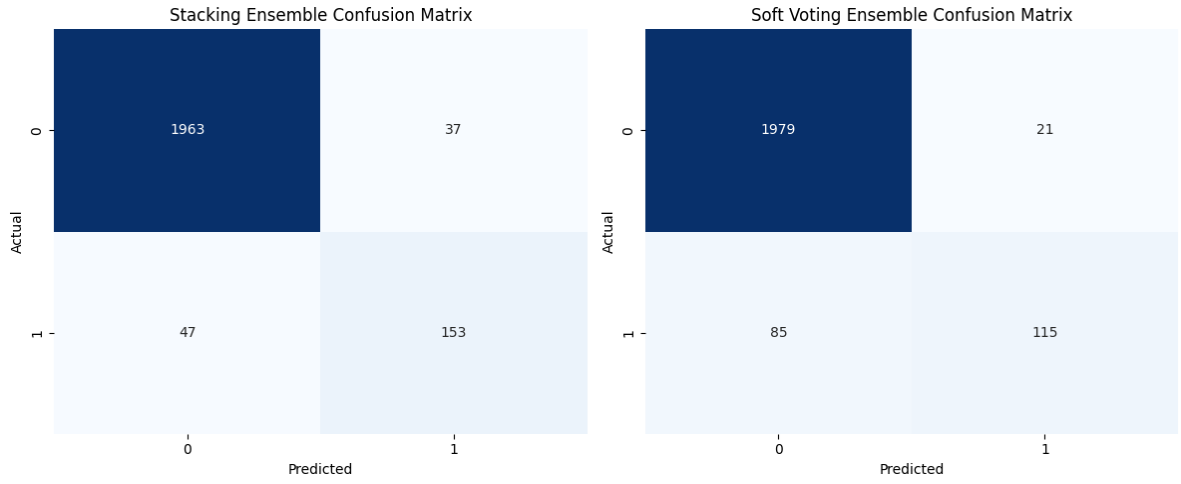

Figure 4: Confusion Matrix - Ensemble Models

6.2 Precision and Recall

Precision and recall are two of the important metrics in phishing detection, where the costs of false positives and false negatives are significant.

6.2.1 Precision

Precision measures the proportion of true positives among all instances predicted as phishing. The Stacking Ensemble achieved a precision of 0.805 and Soft-voting had a precision of 0.845, indicating that it had a high level of confidence in its phishing classifications.

6.2.2 Recall

Recall, or sensitivity, measures the proportion of actual phishing emails correctly identified by the model. The Stacking Ensemble had a recall of 0.765 and soft voting lagged at 0.575.

The following bar chart compares the precision and recall of the models:

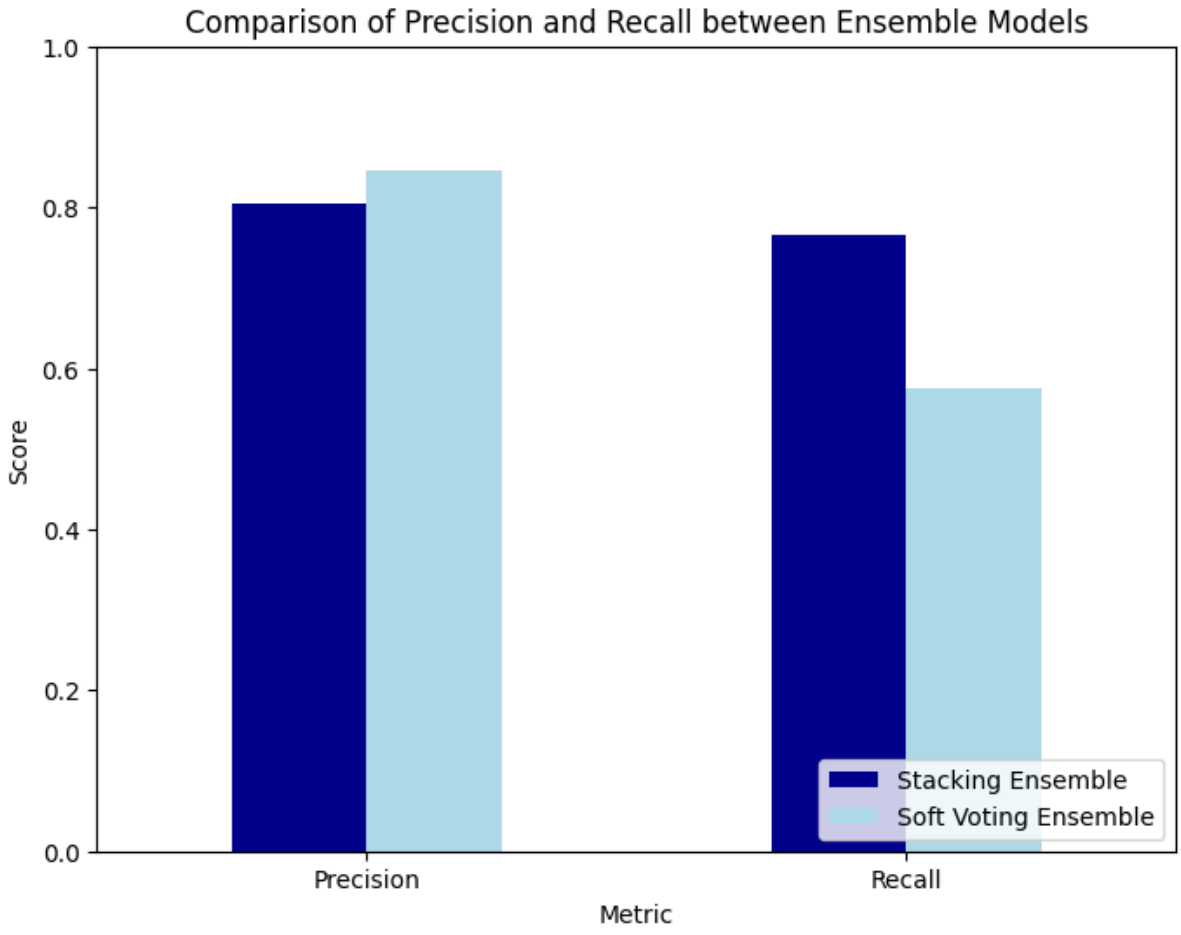

Figure 5: Comparison of Precision and Recall of Ensemble Models

6.3 Advanced Metrics: Matthews Correlation Coefficient (MCC) and AUC

6.3.1 Matthews Correlation Coefficient (MCC)

MCC is a balanced metric that considers all four categories of the confusion matrix (TP, TN, FP, FN). It is particularly useful for evaluating models on imbalanced datasets. The Random Forest model achieved an MCC of 0.898 in content analysis and SVM got an MCC of 0.991 in text-based analysis, indicating a strong correlation between the predicted and actual classifications.

6.3.2 Area Under the Curve (AUC)

The AUC provides a summary of the model's ability to discriminate between positive and negative classes. A higher AUC indicates better model performance. The SVM model achieved an AUC of 0.994 in text-based analysis and RF got an AUC of 0.930, while the Stacking Ensemble and Soft Voting ensembles achieved an AUC of 0.971, both demonstrating superior performance in distinguishing between phishing and legitimate emails across various thresholds.

6.4 Comparative Analysis of Textual and Content-Based Models

6.4.1 Textual Analysis with BERT

The BERT-based textual models, particularly the SVM showed strong performance in understanding the semantic nuances of phishing emails with an accuracy of 99.86% and F1-score of 0.992. This model benefited from BERT's ability to capture context, making them particularly effective in identifying phishing attempts that relied on sophisticated language or deceptive phrasing.

The following figure shows the performance metrics of the 4 different classifiers in textual analysis:

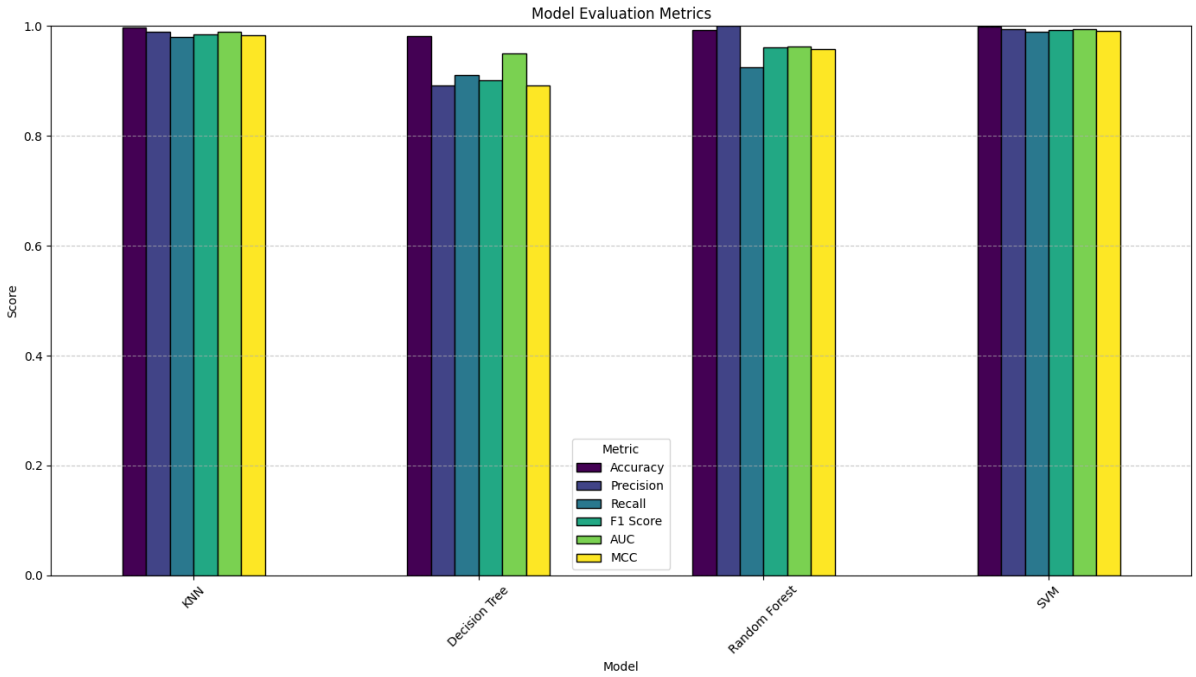

Figure 6: Performance Metrics - KNN, DT, RF, SVM for Textual Analsyis

6.4.2 Content-Based Analysis

Content-based models focused on structural features such as HTML tags, scripts, and hyperlinks. The Random Forest excelled in this area, leveraging these features to detect phishing attempts with high F1-Score of 0.905 and accuracy of 0.983. These models were particularly effective in identifying emails that employed technical elements, such as embedded forms or suspicious URLs.

The following figure represents the performance metrics of the four different classifiers in content based analysis:

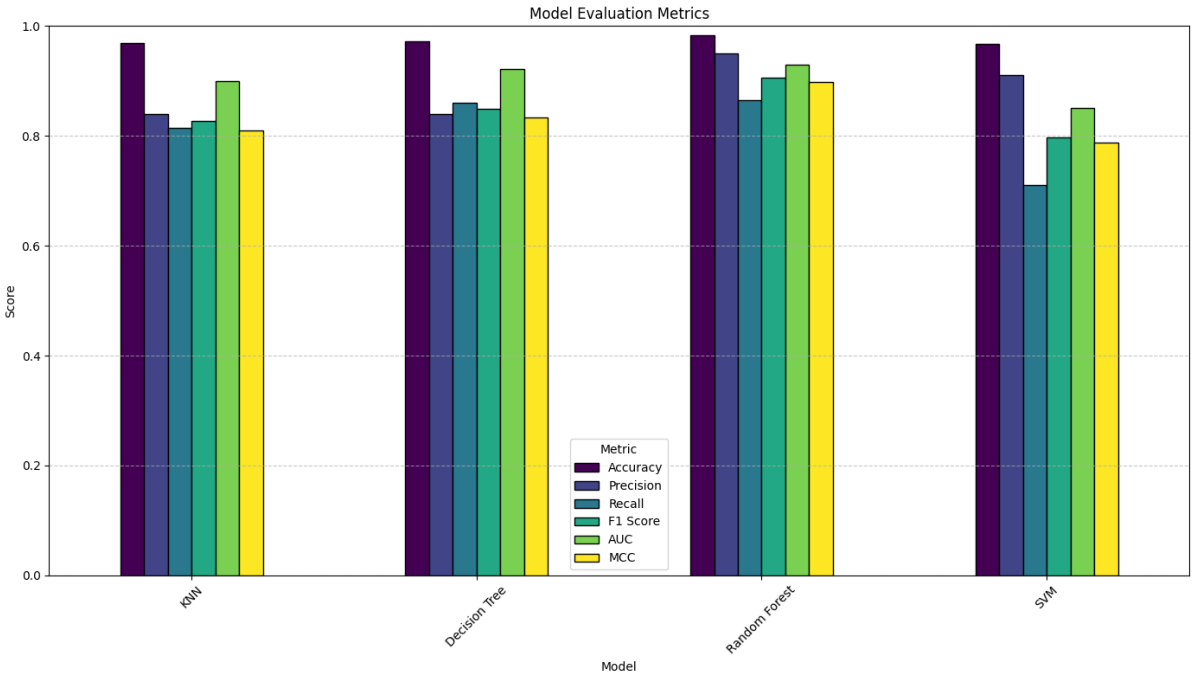

Figure 7: Performance Metrics - KNN, DT, RF, SVM for Content Analysis

6.4.3 Ensemble Models

The combination of textual and content-based models in the Stacking and Soft Voting ensembles provided an accuracy of 96.18% and 95.18%. The figure below shows performance metrices of both the ensemble models.

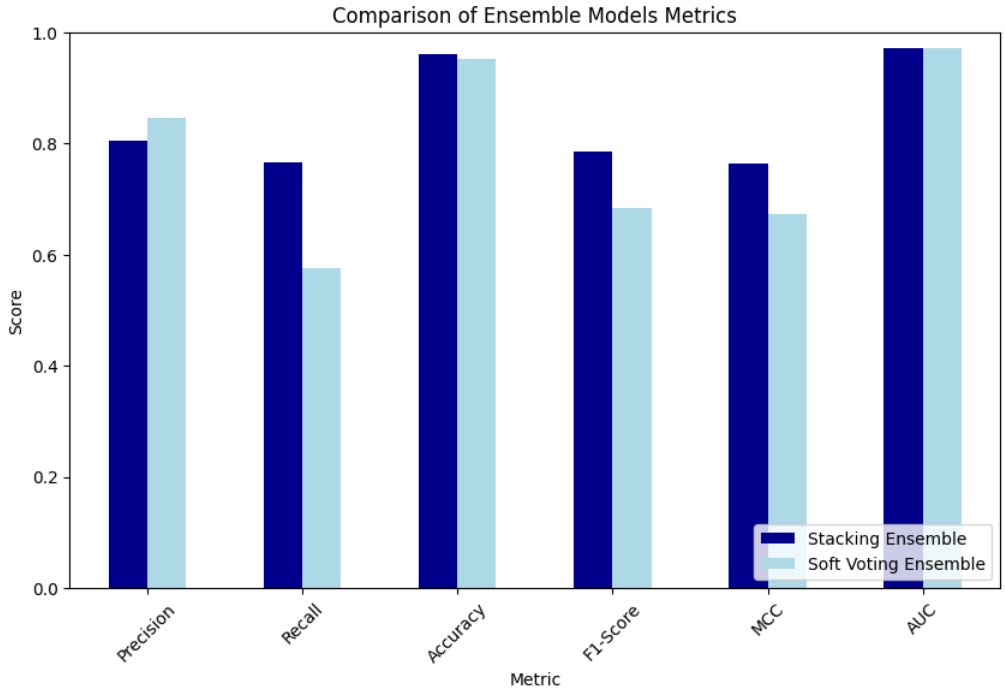

Figure 8: Comparison of Permance Metrics of Ensemble Models

In conclusion, the evaluation of the phishing detection system demonstrates the effectiveness of using a hybrid approach that combines textual and content-based features with advanced machine learning models and ensemble techniques. The study's results provide valuable insights into the strengths and limitations of different models and offer practical guidance for improving phishing detection in real-world settings.

6.5 Discussion

These results from experiments performed in this study provide a detailed evaluation of different machine learning techniques when applied to text and content-based analysis, both individually as well as collectively combined into ensemble models. Our results reveal accuracy, precision, recall F1 score AUC and MCC of different classifiers which help in identifying the strengths and weaknesses of each approach.

In the text analysis, we found that Support Vector Machine (SVM) performed better than other models with an accuracy of 0.9986 and F1 score value of 0.9986. The SVM also had a high MCC (0.9917), indicating good performance for each class. The KNN model, which had an accuracy of 0.9973 and an MCC of 0.9834 performed well but has slightly lower recall (1–2%)than the SVM model. The Random Forest model performed well on Precision but low recall, indicating that it might have missed some true positives.

In content-based analysis the highest accuracy (0.9836) was obtained by Random Forest, and showed the best F1 score of 0.9058, indicating an overall and balance between precision and recall. On the other hand, while SVM performed best in text classification it did pretty bad with a recall of 0.710 (reducing F1 to be as low as 0.7978). This drop is likely due to the nature of features that content-based methods use, and these may not necessarily lend themselves to being well aligned with SVM's capabilities for linear separation. In this case, the KNN model performed more poorly than others in terms of accuracy (0.9691) and MCC score(0.8106), showing that it is a data-sensitive method to apply.

While looking at the ensemble models, both stacking & soft voting ensembles resulted in lower performance compared to few base learners are good enough individually. The stacking ensemble resulted in a precision of 0.8052 and recall of 0.765, leading to an F1 score of 0.7846. Soft Voting on the other hand had a slightly higher precision (0.8456), but lower recall (0.5750), resulting in an F1 score of 0.6845. This hints that the ensemble methods have not done an exact job in combining the base learners, especially recall value is very important in scenarios where false negatives are expensive. The lower MCC values for the ensembles again highlight that finding a balance between true positive and negative could be hard with combined models as some of those individual models performed best in terms of balancing true positives versus negatives, especially SVM.

Although the experimental design was well-founded for taking into account a number of models and ensemble strategies, there are some ways it could have been better. Firstly, the choice of base learners in ensemble methods could be improved by including more diverse models which may translate into better generalisation and a higher diversity among ensembled model performances. Further tuning for the meta-learner in the stacking ensemble or perhaps a change to an even stronger model such as neural network would also be good, because base learners are likely have some interactions and we need more advanced method/models can capture it well. Additionally, a technique to reconcile predictions in the soft voting ensemble which is more advanced than uniform vote aggregation using individual model votes could alleviate this problem of over-devising top-performing models.

The results are therefore consistent with research that has long emphasized the difficulty in constructing successful ensembles, especially when base learners differ and ensemble techniques do not use them to greatest advantage. The decrease in recall observed with the ensemble models is consistent with prior literature that has pointed out how sometimes ensembling can compromise predictions, as it tends to average tendencies across many algorithms possibly leading them to miss on minority classes or subtle patterns of information.

7 Conclusion and Future Work

The main objective of this study is to provide a comparative analysis on how different machine learning models and ensemble learning methods perform in textual and content based analysis. Results showed that, individual models observed in this study, namely SVM and Random Forest achieved high accuracy as well as balanced performance metrics, yet ensemble methods were not able to overcome the best single model. In particular, the stacking and soft voting ensembles struggled to improve recall and MCC which resulted in lower F1 scores than some of their base learners. The results show that the currently ensemble methods are not best performing and hence a new optimized method should be taken for combinig predictions of base learners in better way. In conclusion this study provided an answer at the narrow research question and achieved its objectives, but can also be seen as a starting point to improve systematic review based ensemble models.

Ensemble methods are able to improve the predictions of other classifiers, so they should be a topic for future research. New approaches to these should include the invention of more sophisticated ensemble methods such as those based on weighted voting or boosting that are basically designed to combine multiple base learner predictions much more effectively. The performance of stacking ensembles could be further improved by also including a broader and more varied set of base learners as well optimizing the meta-learner. Research on other feature selection and data preprocessing techniques may also indicate the possibility of increasing model accuracy as well as balance. On the other hand, future studies could test the effect of hyperparameter tuning or different ensemble architectures for this purpose to

eliminate at least some limitations. These endeavors will support a richer, more advanced field of machine learning systems designed to handle difficult analysis operations.

References

Adebowale, M.A., Lwin, K.T. and Hossain, M.A. (2020) 'Intelligent phishing detection scheme using deep learning algorithms', *Journal of Enterprise Information Management*, 36(3), pp. 747–766. Available at: https://doi.org/10.1108/JEIM-01-2020-0036.

Ali, W. and Ahmed, A.A. (2019) 'Hybrid intelligent phishing website prediction using deep neural networks with genetic algorithm-based feature selection and weighting', *IET Information Security*, 13(6), pp. 659–669. Available at: https://doi.org/10.1049/iet-ifs.2019.0006.

Bountakas, P. and Xenakis, C. (2023) 'HELPHED: Hybrid Ensemble Learning PHishing Email Detection', *Journal of Network and Computer Applications*, 210, p. 103545. Available at: https://doi.org/10.1016/j.jnca.2022.103545.

Ding, Y. *et al.* (2019) 'A keyword-based combination approach for detecting phishing webpages', *Computers & Security*, 84, pp. 256–275. Available at: https://doi.org/10.1016/j.cose.2019.03.018.

Fang, Y. *et al.* (2019) 'Phishing Email Detection Using Improved RCNN Model With Multilevel Vectors and Attention Mechanism', *IEEE Access*, 7, pp. 56329–56340. Available at: https://doi.org/10.1109/ACCESS.2019.2913705.

GhaziM.Jameel, N. and E. George, L. (2013) 'Detection of Phishing Emails using Feed Forward Neural Network', *International Journal of Computer Applications*, 77(7), pp. 10–15. Available at: https://doi.org/10.5120/13405-1057.

Gupta, A.Al.B.B. and Manickam, T.C.W.A.A.S. (2013) 'Phishing Dynamic Evolving Neural Fuzzy Framework for Online Detection "Zero-day" Phishing Email', *Indian Journal of Science and Technology*, 6(1), pp. 1–5. Available at: https://doi.org/10.17485/ijst/2013/v6i1.18.

Hamid, I.R.A., Abawajy, J. and Kim, T.-H. (2013) 'Using feature selection and classification scheme for automating phishing email detection'. Available at: https://figshare.utas.edu.au/articles/journal_contribution/Using_feature_selection_and_classification_scheme_for_automating_phishing_email_detection/22908221/1 (Accessed: 11 August 2024).

Harikrishnan, N.B. *et al.* (2019) 'Time Split Based Pre-processing with a Data-Driven Approach for Malicious URL Detection', in A.E. Hassanien and M. Elhoseny (eds) *Cybersecurity and Secure Information Systems: Challenges and Solutions in Smart Environments*. Cham: Springer International Publishing, pp. 43–65. Available at: https://doi.org/10.1007/978-3-030-16837-7_4.

Hota, H.S., Shrivas, A.K. and Hota, R. (2018) 'An Ensemble Model for Detecting Phishing Attack with Proposed Remove-Replace Feature Selection Technique', *Procedia Computer Science*, 132, pp. 900–907. Available at: https://doi.org/10.1016/j.procs.2018.05.103.

Korkmaz, M., Sahingoz, O.K. and Diri, B. (2020) ‘Detection of Phishing Websites by Using Machine Learning-Based URL Analysis’, in *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1–7. Available at: https://doi.org/10.1109/ICCCNT49239.2020.9225561.

Muralidharan, T. and Nissim, N. (2023) ‘Improving malicious email detection through novel designated deep-learning architectures utilizing entire email’, *Neural Networks*, 157, pp. 257–279. Available at: https://doi.org/10.1016/j.neunet.2022.09.002.

Nosseir, A., Nagati, K. and Taj-Eddin, I. (2013) ‘Intelligent Word-Based Spam Filter Detection Using Multi-Neural Networks’, 10(2).

Wei, B. *et al.* (2019) ‘A Deep-Learning-Driven Light-Weight Phishing Detection Sensor’, *Sensors*, 19(19), p. 4258. Available at: https://doi.org/10.3390/s19194258.

Zhu, E. *et al.* (2023) ‘CCBLA: a Lightweight Phishing Detection Model Based on CNN, BiLSTM, and Attention Mechanism’, *Cognitive Computation*, 15(4), pp. 1320–1333. Available at: https://doi.org/10.1007/s12559-022-10024-4.