Improving Phishing Email Detection using Hybrid Machine Learning Approach

Yoga Shri Murti [1] and Palanichamy Naveen [1]

[1] Faculty of Computing and Informatics, Multimedia University (MMU), Cyberjaya, Malaysia.

* Corresponding author. Email: p.naveen@mmu.edu.my

Abstract. Phishing emails pose a severe risk to online users, and their identification is essential to maintaining digital communication security. Phishing email detection techniques are continuously researched due to the evolution of phishing strategies. Machine learning has shown to be a powerful method for automatically identifying phishing emails. However, most of the machine learning methods now in use are based on Support Vector Machines or Naive Bayes, but they are either slow or ineffective at handling the spam filtering problem. This study attempts to provide a phishing email detector and reliable classifier using a hybrid machine classifier with Term Frequency-Inverse Document Frequency (TF-IDF), an effective feature extraction technique. We employ a real-world dataset from Kaggle that has actual ratios of authentic and phishing emails. Next, Exploratory Data Analysis (EDA) is carried out to comprehend the dataset better and spot evident mistakes and outliers to aid the detection process. The feature extraction technique converts the data text into a numerical representation that can be used for machine learning algorithms. The model's performance is evaluated using accuracy, precision, recall, F1-score, receiver operating characteristic curve (ROC), and Area under the ROC curve (AUC) metrics. The study concludes that the hybrid model with TF-IDF feature extraction; achieved high performance with an accuracy of 87.5%. This paper provides significant insight into the application of machine learning for detecting phishing emails and emphasizes the value of mixing different models for enhanced performance.

Keywords: Machine Learning, Phishing email detection, hybrid classification

1. Introduction

In the modern era, email has become the primary mode of official communication, both in government and non-government sectors. It is widely recognized as an effective, fast, and cost-efficient method of communication. However, with the ever-increasing use of email, the risk of receiving unsolicited and malicious content such as spam, fraud, and phishing emails has also risen. According to a study by Kolmar (2022), a staggering 333.2 billion emails are sent worldwide every day, highlighting the high prevalence of email usage and the potential for encountering unwanted content.

In the first quarter of 2023, Vade, a leading email security firm, detected a significant surge in phishing emails. They identified 562.4 million phishing emails, an increase of 284.8 million compared to the previous quarter. This represented a 102% quarter-over-quarter rise, making it the highest Q1 total since 2018 (Stansfield, 2023). These phishing attacks typically begin with victims receiving emails that impersonate trustworthy entities, such as Wallet Connect, a program that links mobile cryptocurrency wallets to decentralized applications. By masquerading as a legitimate source through electronic communication, these emails deceive individuals into divulging sensitive information like passwords, credit card details, and personal data. The email urges recipients to verify their wallets to prevent account suspension.

Machine learning has the potential to revolutionize the detection and prevention of phishing attacks. By leveraging large datasets and advanced algorithms, machine learning offers the promise of more precise and efficient phishing detection compared to traditional methods. This can play a vital role in safeguarding individuals and organizations from the detrimental consequences of phishing attacks.

The research problem in the field of phishing email detection lies in the need to improve the accuracy and efficiency of machine learning algorithms in classifying and detecting phishing emails. While previous studies have explored various methods, including email clustering and traditional detection techniques, the results have not yielded a highly accurate and time-friendly solution. Therefore, there is a crucial need to enhance the performance of machine learning algorithms in effectively identifying and categorizing phishing emails.

The aim of this paper is to propose a novel hybrid machine learning classification approach, to enhance the detection of phishing emails for better protection against associated risks. The study explores the benefits of using multiple combinations of machine learning algorithms to deliver improved performance, increased robustness, enhanced interpretability, and effective real-time predictions.

The subsequent sections of this manuscript are meticulously arranged to provide a comprehensive understanding of the study. Section 2 encompasses some related studies that were conducted in the same field. Section 3 provides a detailed account of the research methodology. The research setup is explained and the outcomes of the study are presented in section 4. Section 5 concludes the paper with a concise and well-organized summary of the entire study.

2. Related Studies

The literature review for this project is a summary or analysis of journals, conference papers, and other sources that are relevant to section 2.1's discussion of phishing email detection using ML algorithms. Section 2.2 contains feature extraction techniques used in reviewed papers. Then measurements in section 2.3 that assess detection performance. The overall discussion of the papers under evaluation is covered in section 2.4.

2.1. Machine learning

In their study, Gallo et al. (2021) used supervised machine learning (ML) to analyze suspicious emails for phishing attempts. The project faced challenges such as evaluating all received emails, requiring human intervention, lacking protection against individual victim attacks, and limitations of supervised learning. To assess performance, the authors relied on manual identification of reported emails, making it infeasible for unreported ones. They evaluated various ML algorithms, including NB, Nearest Neighbour's, Linear SVM, RBF SVM, DT, RF, AdaBoost, and MLP Neural Net, based on precision, recall, and F-score. The results showed that RF, utilizing 36 characteristics, achieved a maximum precision of 95.2%, recall of 91.6%, and F-score of 93.3%. The authors suggested future research explore unsupervised ML for the same objective.

Furthermore, author of (KARIM et al., 2020) proposes an anti-spam framework which evaluates based on domain and header of emails headers. They worked with six different clustering algorithms and concluded that their proposed approach named 'Optics' resulted an average of 3.5% better efficiency compared than spectral and k-means algorithm. Study by Vijaya et al. (2019), different classification and regression methods were implemented on an email spam filter dataset, with a specific focus on identifying spear phishing methods. Ensemble methods such as boosting, nagging, stacking, and voting were incorporated into existing algorithms to improve the classification results. Through evaluation, they discovered that the KNN algorithm provided efficient predictions and effective implementation among the available algorithm.

In the paper (Jawale et al., 2018) proposed a hybrid spam filtering algorithm which has more accuracy when combining both filter model. NB has faster classification speed but requires small dataset and has low accuracy performance. The SVM has the highest accuracy performance, slow classification speed, and requires large dataset. This paper Implemented NB and SVM together to utilize both algorithms' advantages and to minimize the drawbacks. This combination achieves 99.44% accuracy which is higher compared to the result from implementing this algorithm separately. Recall and precision ration is used to measure the spam filtering performance. The spam detection by separate algorithm has been measured to compare the performance with combining proposed hybrid spam detection algorithm.

Next, Study (Vazhayil et al., 2018) focuses on developing phishing detection models through a non-sequential approach using classical ML classifications. The pre-processed data is converted by Term Document Matrix into a numeric values (TDM). The numerical data is subsequently fed ML algorithms, such as DT, KNN, LR, NB, RF, AdaBoost, and SVM. Despite the study's unbalanced dataset, Random Forest outperformed other methods of ML in terms of classification accuracy, leading to high categorization rates. The authors also note that integrating data from outside source can readily improve the suggested phishing detection architecture without relying on feature selection, which necessitates domain expertise.

The paper (RAZA et al., 2022) focuses on the different techniques for classifying spam emails using ML algorithms. The authors emphasize that a supervised ML approach is the most adopted method, with the research mainly focused on bag of words (BOW) and body text features. The paper highlights the need for focusing on different features, the use of multi-algorithm systems, real-time spam classification, and small false positive rates. The authors found that multi-algorithm systems tend to perform better than using a single algorithm, with algorithms such as NB and SVM being the most commonly used ML algorithms in this field.

Also, paper (Yuliya et al., 2020) the authors used a readymade dataset of 5728 emails from Kaggle, which consisted of both spam and non-spam (ham) emails. The authors used tokenization to split the sentences into words and Count Vectorizer methods to convert words to numbers for feature extraction. The authors employed f-measure, accuracy, precision, recall, and the ROC area to assess the model quality. They used 6 alternative models to determine if an email was spam or not, and all but the Naive Bayesian classifier underwent hyperparameter optimization using GridSearchCV. Although Logistic Regression was the most effective algorithm, the Naive Bayesian classifier had the highest level of accuracy, reaching 99%.

In another paper by Adi et al. (2016), a hybrid model combining Logistic Regression and Decision Tree was proposed for spam email classification. Logistic Regression was used to filter noisy data before feeding it to the Decision Tree, as Decision Trees are sensitive to noise and can perform poorly in its presence. A false negative threshold was introduced to enhance true positive results and improve Decision Tree induction. The hybrid model (LRFNT+DT) achieved an accuracy of 91.7% on a spam base dataset containing 57 attributes and 4061 messages. The authors concluded that Logistic Regression can enhance the performance of Decision Trees by mitigating the impact of noisy data.

Lastly, Lew et al. (2022) proposed a method for detecting phishing emails using hybrid features and compared it to the Hybrid Approach (HA) from a previous study. The method utilized 9 features, including domain sender, blacklist words, IP address, symbols, and unique sender, selected based on common phishing techniques. Support Vector Machine (SVM) was employed as the classifier. The proposed method outperformed the HA approach with a higher accuracy of 97.25% and a lower error rate of 2.75%, compared to 95.50% accuracy and 4.50% error rate for HA. However, the proposed method was noted to be time-consuming and had limitations with the blacklist keyword feature.

Table 1. Summary of ML classifier used

No.	Authors	RF	DT	SVM	NB	LR	K-Means	KNN	DL
1.	(Gallo et al., 2021)	/	/						
2.	(Karim et al., 2020)						/		
3.	(Fergus et al., 2022)		/						

4.	(Raza et al., 2022)		/	/	/		/	/	
5.	(Fang et al., 2019)								/
6.	(Vazhayil et al., 2018)	/	/	/	/	/		/	
7.	(Vijaya et al., 2019)	/		/	/			/	
8.	(Yuliya et al., 2020)	/	/	/	/	/		/	
9.	(Doaa et al., 2020)		/		/				
10.	(Jawale et al., 2018)			/	/				
11.	(Adi et al., 2016)		/			/			
12.	(Lew et al ., 2022)			/					

2.2. Feature extraction techniques

Concisely, Table 2 provides a summary of the authors' feature extraction/selection techniques used in the reviewed research papers. Along with the commonly employed techniques, there are some additional methods mentioned as well. Among the papers listed in Table 2, TF-IDF was the most frequently used feature extraction technique, employed by five authors (Gallo et al., 2021; Karim et al., 2020; Vazhayil et al., 2018; Vijaya et al., 2019; Adi et al., 2016). Two papers utilized both Bag of Words (BoW) techniques Raza et al., 2022 and Fergus et al., 2022. Count Vectorization (CV) was used by three authors: Jawale et al., 2018; Lew et al ., 2022 and Yuliya et al., 2020. Information Gain (IG) was employed by three papers: Fang et al., 2019; Vazhayil et al., 2018; and Jawale et al., 2018. One paper, Raza et al., 2022, did not specify any particular feature extraction or selection technique. Information Gain (IG) was employed by four papers: Fergus et al., 2022; Fang et al., 2019; Doaa et al., 2020; and Jawale et al., 2018. In addition to the mentioned techniques, several authors used alternative methods. Gallo et al. 2021; employed a wrapper method, while Karim et al. 2020 utilized Principal Component Analysis (PCA), Laplacian Score, and Multi-Cluster-based Feature Selection (MCFS). Lew et al. 2022 incorporated a hybrid feature selection approach involving blacklist keywords. Vazhayil et al. 2018 explored Term-Document Matrix (TDM) along with Singular Value Decomposition (SVD) and Non-Negative Matrix Factorization (NMF) as feature extraction methods. Adi et al. 2016 utilized a threshold-based approach for feature selection. Overall, the most widely used feature extraction technique among the papers in Table 2 was TF-IDF, while Count Vectorization (CV) and Information Gain (IG) were also commonly employed. Some authors utilized additional techniques such as wrapper methods, PCA, Laplacian Score, MCFS, hybrid feature selection, and threshold-based selection.

Table 2. summarizes the authors' feature extraction/selection techniques.

No.	Authors	TF-IDF	BoW	CV	IG	Other
1.	(Gallo et al., 2021)	/				wrapper
2.	(Karim et al., 2020)	/				PCA Laplacian MCFS
3.	(Fergus et al., 2022)		/		/	
4.	(Raza et al., 2022)		/			
5.	(Fang et al., 2019)				/	
6.	(Vazhayil et al., 2018)	/				TDM SVD NMF

7.	(Vijaya et al., 2019)	/				
8.	(Yuliya et al., 2020)			/		
9.	(Doaa et al., 2020)				/	
10.	(Jawale et al., 2018)			/	/	
11.	(Adi et al., 2016)	/				threshold
12.	(Lew et al ., 2022)			/		hybrid feature blacklist keywords

2.3. Performance metrics

Table 3 summarizes evaluation metrics that are highly used in related research papers. Among the listed papers, several evaluation metrics were commonly used. Accuracy, Precision, Recall, and F-measure were the most frequently employed metrics, utilized by multiple authors, including Fang et al., 2019; Vazhayil et al., 2018; Vijaya et al., 2019; Yuliya et al., 2020; Doaa et al., 2020; and Lew et al., 2022. Confusion Matrix was used by Fang et al., 2019; Vijaya et al., 2019; and Adi et al., 2016, while AUC and ROC were employed by Vijaya et al., 2019; and Lew et al., 2022.The table highlights the common evaluation trend of focusing on metrics related to accuracy, precision, recall, and F-measure, which are essential for assessing the performance of phishing email detection models. These metrics provide insights into the effectiveness of the models in correctly classifying phishing emails and minimizing false positives and false negatives.

Table 3. summary of evaluation metrics

No.	Authors	Accuracy	Precision	Recall	F-measure	Confusion Matrix	AUC	ROC
1	(Gallo et al., 2021)		/	/	/		/	
2	(Karim et al., 2020)	/						
3	(Fergus et al., 2022)			/				
4	(Raza et al., 2022)	/						
5	(Fang et al., 2019)	/	/	/	/	/		
6	(Vazhayil et al., 2018)	/	/	/	/			
7	(Vijaya et al., 2019)	/	/	/	/		/	/
8	(Yuliya et al., 2020)	/	/	/	/			/
9	(Doaa et al., 2020)		/	/	/	/		
10	(Jawale et al., 2018)		/	/				
11.	(Adi et al., 2016)				/	/		
12.	(Lew et al ., 2022)	/		/		/		

2.4. Discussion of reviewed papers

Some restrictions on the use of learning algorithms for phishing detection have indeed been found in the review of recent literature. The complexity of phishing emails, which are designed to look like legitimate emails, presents challenges for accurate classification. This project aims to find the efficient individual algorithms which has high accuracy of phishing detection and then combine to form an effective hybrid algorithm. The machine learning literature review shows that one of the drawbacks of using Support Vector Machines (SVM) for phishing detection is that it can be time-consuming to train and test the models. However, the combination of Naive Bayes and SVM has proven to be effective for spam classification compared to other machine learning methods as in (Jawale et al., 2018) researchers implemented this NB and SVM together to utilize both algorithms' advantages and to minimize the drawbacks. This combination achieves 99.44% accuracy which is higher compared to the result from implementing this algorithm separately. Most of the prior studies have used decision trees, Naive Bayes, SVM, and random forest for experiments. There have been a few papers that used deep learning, such as Recurrent Convolutional Neural Networks (RCNN) as in the study by (Fang et al., 2019), which achieved high accuracy. Naive Bayes, random forest, and decision trees are the most used and recommended machine learning methods in prior studies, as they perform well and are efficient in terms of time.

Based on feature extraction /selection techniques most research in the field has utilized feature extraction methods such as TF-IDF, bag of words, and count vectorization. However, there was a different approach, such as the hybrid feature extraction approach described in (Lew et al ., 2022) , which also had limitations such as blacklist keywords and time consumption. As a result, this project will use TF-IDF and count vectorization as the feature extraction techniques. For performance evaluation metric, accuracy, precision, recall, and F1 score are regularly used assessment metrics in reviewed papers, and as well as well as in this paper. The performance will also be visualized using ROC and AUC measurements. The other limitations were that most of the researchers used spam base dataset to conduct their projects because phishing email is not concerned enough for experiments. As a result, this project focuses on using phishing email datasets for all the experiments.

3. Research Methodology

The overall flow of the proposed approach is shown in Fig. 1. The proposed flow provides a systematic and structured framework for building and evaluating machine learning classifier to detects phishing emails, which are described Section 3.1 – 3.8.

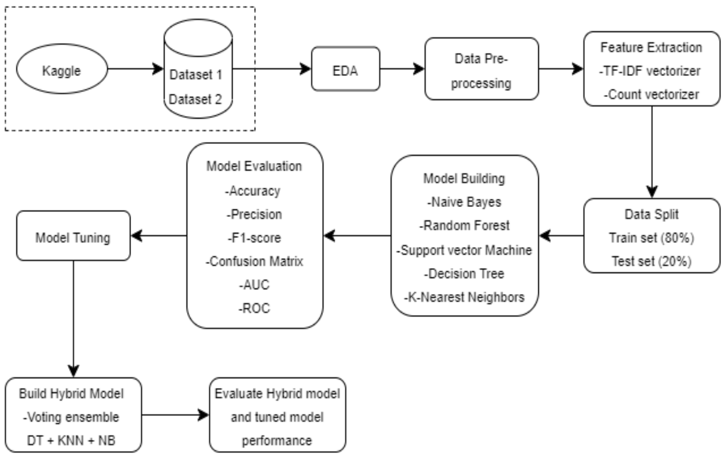

Figure 1 Proposed Project Methodology

3.1. Data collection

The dataset used for this project was obtained from Kaggle, a platform that allows users to find, publish, and explore datasets in a web-based data science environment. Two types of phishing datasets were collected, one being a large unbalanced and pre-made dataset, and the other being a balanced raw phishing email dataset. Dataset 1 (*Phishing Email Collection*, n.d.) was used to compare the performance of machine learning algorithms with those previously used machine learning algorithms in reviewed paper. The Dataset 2 (*Phishing Email Data by Type*, n.d.) was used to construct a hybrid phishing detection model using the proposed method in this project.

3.2. Exploratory data analysis

EDA techniques are carried out for dataset to have a look at data before continuing use for next steps. EDA helps to identify obvious errors, detects outliers, and gives better understanding about relations among the variables. Dataset 1 is unbalanced and dataset 2 is balanced. EDA helps to drop Unwanted columns. The targeted columns were renamed. Then label encoded were used to encode the targeted column such as Phishing emails labelled as 1 and legitimate emails labelled as 0. Missing null values and duplicated values in the dataset are removed. The email texts were tokenized and grouped according to the total number of characters, number of words and number of sentences.

3.3. Data pre-processing

Datasets were in csv file format. To perform data pre-processing, the needed libraries must import. Libraries like NumPy, pandas, Nltk, sklearn, matplotlib and other necessary machine learning model libraries . Next feature scaling and feature extraction steps conducted to prepare the model to perfume effectively without noises in the dataset. The feature extraction steps are as follows:

3.3.1. Lower case

Converting all the characters to Lowercase gets rid of unimportant parts of the data or noise. This approach maintains consistent flow during the mining and for natural language processing tasks. Lower casting all the texts makes the whole data straightforward.

3.3.2. Tokenization

Tokenization breaks or splits larger pieces of text into smaller units called tokens. It is an important step in natural language processing (NLP) because it enables the creation of useful representation of text data to analyse and manipulate. The type of tokenization used in this project is Word tokenization is where text is broken down into individual words.

3.3.3. Removing special characters

Removing special characters from text is a common pre-processing step in NLP analysis. Special characters are punctuation marks, symbols and other alphanumeric characters that do not gives a significant meaning of the text. By removing these characters, the text can be cleaned and made more suitable for further processing.

3.3.4. Removing stop words and punctuation

English Stop words are common words in a language that little meaning on their own and are often used to connect other words in a sentence. Few of the stop words are “the”, “an “, “and” and” They are removed from a text corpus because they don’t contribute much meaning to the text and slows down the processing.

3.3.5. Stemming

This is a process that reduces words to their base or root form, called a stem. The purpose of stemming is to reduce words while retaining the meaning and to make the analyses process easier when comparing the words in a corpus of text. Porter stemming algorithms is used in this project. Porter stemming is widely used for text pre-processing in NLP and text analysis operation.

3.4. Feature extraction

This method's objective is to transform unstructured text input into a numerical representation that machine learning algorithms can comprehend and use. The two most often used approaches for extracting features from a dataset of phishing emails are TF-IDF and CV.

3.4.1. Count vectorizer

Count Vectorizer, on the other hand, is a simple technique that converts text data into a numerical representation based on the frequency of words in a document. Each document is represented by a row, and each vocabulary term by a column, in a matrix. The values in the matrix represent the count of each word in each document. Table 4 shows how count vectorization looks in theory and table 5represents how count vectorization is done in practical form (Ganesan, 2019). The sentence is “cents, two cents, old cents, new cents: all about money.

Table 4 (in theory)

	about	all	cent	money	new	old	two
doc	1	1	4	1	1	1	1

Table 5 (in practice)

	0	1	2	3	4	5	6
doc	1	1	4	1	1	1	1

3.4.2. TF-IDF

TF-IDF is a powerful tool by Karen Spärck Jones; A well-known method for information retrieval and natural language processing (NLP) for evaluating the significance of a word to a particular document when compared to a collection of documents. It considers both the frequency of the word in the document, referred to as the term frequency, and the inverse document frequency, which indicates how often the word appears in the collection. The combination of these two measures provides the TF-IDF score, which can be used to assess the relevance of a word to a specific document.

The quantity of times a term or syllable appears in a document relative to all the other terms in the document is known as the term frequency (TF) of that term or word. The tf-idf equation is displayed in equation 1.

$$TF = \frac{\text{number of times the term appears in the document}}{Total\ number\ of\ terms\ in\ the\ document} \quad (1)$$

How frequently a word appears across the whole corpus of documents that contain the phrase is determined by its inverse document frequency (IDF). IDF is determined by dividing the number of documents containing the phrase by the logarithm of the total number of documents. The equation to compute IDF is shown in equation 2.

$$IDF = log(\frac{number\ of\ the\ documents\ in\ the\ corpus}{Number\ of\ documents\ in\ the\ corpus\ contain\ the\ term}) \quad (2)$$

The last step in calculating the TF-IDF value for each word within the document is the TF-IDF score. TF and IDF ratings for each syllable are multiplied to obtain the TF-IDF of a term. This number shows the word's relative importance in each document relative to the total corpus of documents, or r. Equation 3 displays the TD-IDF equation.

$$TF - IDF = TF * IDF \tag{3}$$

3.5. Splitting the dataset.

The dataset is divided into two parts, with 80% designated as the training set and 20% as the testing set. This split is commonly used in machine learning because it uses a large enough sample for training the model and separate, independent samples for evaluating its performance. This split also provides a measure how well the model will generalize to new and unseen data using the captured relation patterns during training.

3.6. Machine learning algorithms

This section implements the NB, SVM, RF, DT, and KNN machine learning classifiers. The models are discussed in Sections 3.6.1 to 3.6.5, respectively.

3.6.1. Naïve bayes

The Naive Bayes algorithm based on (What Is Naïve Bayes | IBM, n.d.), calculates the probability of each feature given each class and uses these probabilities to determine the likelihood of each feature set for each class label. It assumes that all features are independent of each other when given the class label, making the likelihood calculation simpler. This algorithm assumes that the features follow a Gaussian distribution and is commonly used for tasks like text classification and phishing detection. Before tokenizing the texts, the dataset is cleaned using NB techniques. The probability of each word (A) is then determined, and the likelihood of an email is represented by equation (4).

$$P(A) = \frac{\frac{a\text{^}spam}{s}}{W(\frac{a\text{^}ham}{h} + \frac{a\text{^}spam}{s})} \tag{4}$$

Where, P(A): is a probability for word.

a^spam: Appearance time of word in spam mail.

a ^ ham: Appearance time of word in spam mail.

s: Number of spam mail.

h: Number of ham mail.

w: weight i.e., tfidf, which is calculated by taking the probability of spam to ham.

The composite probability for the message is then calculated after the spam probability, P(A1), has now been determined.

3.6.2. Support vector machine

Support Vector Machine (SVM) is a popular machine learning method (*SVM Machine Learning Tutorial – What Is the Support Vector Machine Algorithm, Explained with Code Examples*, 2020) ; utilized for both classification and regression tasks. Its main objective is to create a hyperplane, which is a boundary that separates data points into distinct groups and assigns each group to a specific class. The hyperplane is represented by a line, as follows in equation (5) and (6):

$$y = a * x + b \tag{5}$$

$$a.x + b - y = 0 \tag{6}$$

Assume that X = (x, y) and W = (a, 1). In vector, we create a hyperplane as in equation (7):

$$W * X + b = 0 \tag{7}$$

In the context of this study, where x represents the input characteristics, the weight value is denoted by W, and the bias term is represented by b. The study employs a linear kernel Support Vector Machine (SVM), which aims to find the optimal hyperplane for separating opinion data into two binary classes. The process of identifying the ideal hyperplane involves considering the outermost data points from the two classes and incorporating them into the determination of the best hyperplane.

3.6.3. Random forest

Random Forest (RF) is a popular ensemble learning algorithm used for regression and classification tasks referred from (IBM, n.d.) . It combines multiple decision trees to make predictions, resulting in more accurate results. RF can also assess feature importance, aiding in feature selection. However, RF's training phase can be computationally expensive due to the creation of multiple decision trees. In RF and other decision tree-based algorithms, the Gini impurity index measures the impurity within a set of instances based on their class labels. The goal is to find the feature that minimizes the Gini index, indicating higher homogeneity among class labels within the instances. Mathematically, the Gini impurity index can be calculated using equation (8).

$$\begin{aligned} \text{GINI INDEX} &= 1 - \textstyle\sum_{i=1}^{n}(Pi))^2 \\ &= 1 - [(P+)^2 + (P-)^2] \end{aligned} \tag{8}$$

3.6.4. Decision tree

A Decision Tree is a versatile algorithm commonly employed in various applications such as diagnosis as in (Saini, 2021) , segmentation, and phishing risk assessment. It uses a tree-like structure and a series of rules to predict the value of a target variable. Each node in the tree represents a test on an input feature, with branches representing possible outcomes. The leaves of the tree correspond to the model's final predictions. The tree is built by repeatedly dividing the data into smaller subsets based on the most influential features for the target variable. This process equation (9) continues until the subset is homogeneous, meaning that all instances in the subset have the same target variable value.

$$\mathrm{E(S)} = \textstyle\sum_{i=1}^{c} - \; pi\, log2 \text{ pi} \tag{9}$$

3.6.5. K-nearest neighbours

KNN (K-Nearest Neighbors) classification (Harrison, 2018), predicts the class label of a new sample based on the class labels of its K nearest neighbors in the training data. The value of K, determined by the user, determines the number of neighbors considered. KNN makes no assumptions about the data distribution and is easy to use. However, it takes longer to make predictions with larger datasets since it requires finding the K nearest neighbors for each new sample. The algorithm calculates the distance between the new data point and all other instances using a chosen distance metric, such as Euclidean distance, and assigns the class label based on the majority vote among the K most similar neighbors. The Euclidean method, a standard distance measurement technique, is shown in equation (10).

$$\mathrm{d(x,y)} = \sqrt{\textstyle\sum_{i=1}^{n}(yi - xi)^2} \tag{10}$$

3.7. Performance evaluation

This study evaluates the model's performance using confusion matrix, accuracy, precision, recall, F1-score, roc curve and auc. The measures are discussed in Sections 3.7.1 to 3.7.7, respectively.

3.7.1. Confusion matrix

An algorithm's performance can be seen using a particular table structure called a confusion matrix, also known as an error matrix by Karl Pearson. This type of method is commonly one for supervised methods (in unsupervised learning usually called a matching matrix). The examples in a predicted class are represented by each column of the matrix, whereas the occurrences in an actual class are represented by each row. In equation (11), the confusion matrix is depicted as false positives (FP), false negatives (FN), true positives (TP), and true negatives (TN). TP + TN + FP + FN represent the total number of tuples. Table 3. Shows the confusion matrix.

	Predicted Positive	Predicted Negative
Actual Positive	TP (True Positive)	FN (False Negative)
Actual Negative	FP (False Positive)	TN (True Negative)

(11)

Where:

- ✓ True positives (TP) are positive tuples that the classifier successfully categorized; TP represent the quantity of real positives.
- ✓ True negatives (TN) are the negative tuples that the classifier correctly categorized; the total number of genuine negatives is TN.
- ✓ False positives (FP) are negative tuples that were mistakenly classified as positive; the number of false positives, FP, shall be used.
- ✓ False negatives (FN) are positive tuples that were incorrectly classified as negatives. The number of false negatives, FN, shall be used.

3.7.2. Accuracy

Accuracy gauges how well an algorithm predicts actual values. The equation (12) derived from (Wikipedia Contributors, 2019).

$$Accuracy = \frac{TP+TN}{(TP+TN+FP+FN)} \tag{12}$$

3.7.3. Precision

Precision is used to compare the purity of the anticipated TP to the TP of the ground truth. The precision is calculated in equation (13) derived from (Wikipedia Contributors, 2019), is particularly relevant when dealing with imbalanced datasets, where the minority class holds more significance or interest.

$$Precision = \frac{TP}{(TP+FP)} \tag{13}$$

3.7.4. Recall

Recall is a metric that assesses how accurately a model makes positive predictions. Out of all the positive cases, the number of accurate positive predictions made by the model is calculated. Recall reveals the situations in which the model incorrectly identified affirmative cases, as in equation (14).

$$Recall = \frac{TP}{(TP+FN)} \quad (14)$$

3.7.5. F1-score

The evaluation statistic known as the (*F-Score*, 2021) is produced using the harmonic mean of recall and precision; which calculated as in equation (15).

$$F1 - score = \frac{2*P*R}{P+ R} \quad (15)$$

Where:
P = precision
R = recall

3.7.6. ROC curve

The accuracy of a binary classifier is visually represented by the Receiver Operating Characteristic (ROC) curve. This technique based on (Narkhede, 2018) is used to show how well a classifier can distinguish between favourable and unpleasant instances. On the x- and y-axes of the ROC curve, the False Positive Rate (FPR) and True Positive Rate (TPR) are displayed, respectively. The TPR and FPR can be evaluated using the formulas in equation (16).

$$TPR = \frac{TP}{(TP+FN)} \qquad FPR = \frac{FP}{(FP+TN)} \quad (16)$$

3.7.7. AUC

The area under the curve is a metric of how effectively a binary classifier can differentiate between two classes (AUC). It provides a clear explanation from (Narkhede, 2018) of the Receiver Operating Characteristic (ROC) curve, a graph that displays how well the classifier performs. AUC is computed by applying the equation shown in equation (17). It offers a lone number that summarizes the classifier's propensity to discern between positive and negative samples, and it is frequently employed in assessing machine learning models.

$$AUC = \frac{TPR+TPR}{2} \quad (17)$$

3.8 Soft voting classifier

The voting classifier (soft) approach used for a hybrid combination model is a machine learning technique that combines the predictions of multiple individual models to make a final decision. In this case, soft voting approach, which means that each model's prediction is assigned a probability or confidence score. These scores are then combined using a weighted average, taking into account the models' performance and reliability. By considering the confidence scores, the method can provide more nuanced and accurate predictions compared to a simple majority voting scheme. This hybrid combination model combines the strengths of multiple models, leveraging their diverse perspectives and expertise, to enhance the overall predictive power and generalization capabilities of this classifier.

4. Evaluation of Findings and Discussion

This section displays the results that were achieved by proposed methods. Section 4.1 discusses the time taken to test and train along with the accuracy and precision of the algorithms. Section 4.2 evaluation of tf-idf and count vectorizer as feature extraction. Section 4.3 shows the performance result of hybrid and tuned hybrid model. Overall, the evaluation of findings provides valuable information that can be used to make informed decisions in the machine learning process.

4.1. Results of machine learning algorithms

Table 6 shows the results of the comparison between different machine learning techniques used for phishing detection. The models were trained and tested on dataset 1, and the time taken for both training and testing was recorded. The train accuracy and test accuracy of each model are also reported. Considering the result shown in the table below, DT model has the fastest training and testing time. It also achieved a high training accuracy of 100% and a respectable test accuracy of 99.104%. Therefore, DT appears to be the fastest and most efficient model among the other models. Following that, RF performs well in terms of training and testing accuracy. While KNN and SVM has high accuracy but the training and testing time for these two models are significantly longer compared to DT and RF.

Table 6 time taken for train and test model with its accuracy

Model	**Time taken to train**	**Time taken to test**	**Train accuracy**	**Test accuracy**
NB	0.2 s	0.1 s	96.754	96.669
KNN	0.5 s	33.9s	98.576	98.316
RF	50.4 s	1.1 s	99.999	99.451
SVM	3m 59.5s	2m 43.9s	98.187	98.130
DT	2.9 s	0.0s	100.0	99.104

Based on table 7, in terms of misclassification rates all the models have relatively low rates, ranging from 0.55% for RF to 3.331% for NB. This indicates that the models can minimize the number of instances that are classified incorrectly.

Table 7. detection's accurate and misclassified rate

ML	Accurate rate %	Misclassified rate %
NB	96.669	3.331
SVM	98.13	1.87
RF	99.45	0.55
DT	99.093	0.907
KNN	98.316	1.684

Figure 2 shows the visualization of model's performance in accuracy and precision. NB and SVM have the least precision compared to the other three models. Accuracy wise all model performed greatly and the most well performed is RF.

.

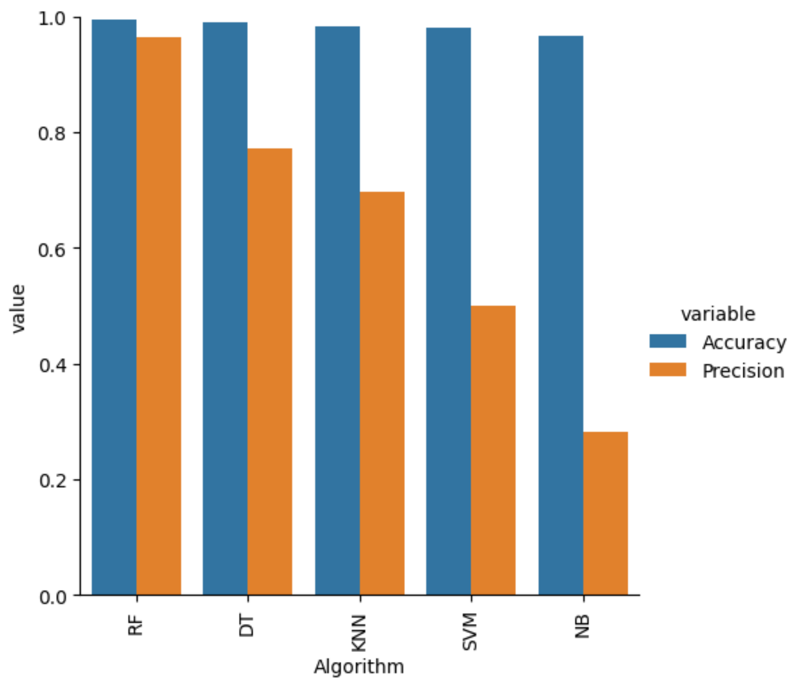

Fig 2 Machine Learning classification algorithms accuracy and precision.

4.2. Results of tf-idf vectorizer and count vectorizer

Based on the provided table, we can observe the performance of different machine learning algorithms on two feature representations: TF-IDF vectorization and Count vectorization. The results highlight the variation in performance based on the feature representation used. Naive Bayes (NB) consistently performs well with both feature representations, achieving high accuracy and precision scores (1.0) in all cases. K-Nearest Neighbors (KNN) and Support Vector Machine (SVM)) performs better in terms of accuracy with TF-IDF compared to Count vectorization. Random Forest (RF) demonstrates similar accuracy scores (0.750) with both feature representations. The precision score is slightly higher (0.700) with Count vectorization compared to TF-IDF (0.6667). Only Decision Tree (DT) exhibits slightly higher accuracy with Count vectorization among all other models. The choice of feature representation (TF-IDF or Count vectorization) does impact the performance of machine learning algorithms. Based on the results presented in Table 8, TF-IDF vectorization demonstrates better performance for phishing email detection compared to Count vectorization. This aligns with the general preference for TF-IDF in text analysis tasks, where the emphasis is on capturing the importance and distinctiveness of terms. Therefore, considering the table results and the overall advantages of TF-IDF in highlighting important terms, TF-IDF is the more preferred choice for phishing email detection and other text analysis tasks.

Table 8. feature extraction accuracy and precision.

Model	**Accuracy TF- IDF**	**Accuracy CV**	**Precision TF -IDF**	**Precision CV**
NB	0.9375	0.875	1.000	1.000
KNN	0.813	0.500	1.000	0.500
RF	0.750	0.750	0.6667	0.700

SVM	0.750	0.625	1.000	0.583
DT	0.6875	0.750	0.800	0.750

4.3. Result of proposed model

The proposed hybrid model combines three algorithms: hyperparameter tuned of Naive Bayes, Decision Tree, and K-Nearest Neighbors, with TF-IDF vectorization. Table 9 shows the hybrid combination algorithms before and after hyperparameter tuning.

By combining the NB, KNN, and DT algorithm, the model offers several advantages. These algorithms can leverage their individual strengths and mitigate their weaknesses. Decision Tree can capture complex relationships and provide interpretability. K-Nearest Neighbors can identify similar instances and benefit from the local patterns in the data. Naive Bayes can model probabilistic dependencies between features and the target variable. Therefore, this combination potentially achieves a more robustness, interpretability, and efficiency of these algorithms, collectively improving the overall performance in detecting phishing emails.

Before hyperparameter tuning, the hybrid model achieves an accuracy of 0.8125, indicating the proportion of correctly classified instances. The precision score is 1.0, reflecting the model's ability to correctly identify positive instances. The recall score is 0.625, representing the model's ability to correctly identify all positive instances. The F1-score, which combines precision and recall, is 0. 769..

After hyperparameter tuning, the hybrid model exhibits significant improvement in performance. The accuracy increases to 0.938, indicating a higher proportion of correctly classified instances. The precision score remains at 1.0, demonstrating the model's consistent ability to correctly identify positive instances. The recall score improves to 0.875, indicating a higher rate of correctly identifying all positive instances. The F1-score shows a notable improvement, reaching 0.933, which indicates a better balance between precision and recall.

Table 9 Proposed hybrid model performance

Model	Accuracy	Precision	Recall	F1-score
NB+DT+KNN	0.8125	1.0	0.625	0.769
Tuned Hybrid	0.938	1.0	0.875	0.933

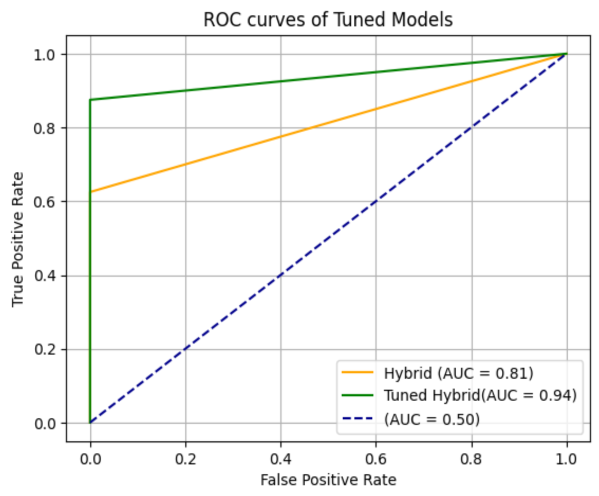

Figure 3 shows the AUC score and ROC curve of hybrid model and tuned hybrid model.

Overall, the hyperparameter tuning process enhances the performance of the hybrid model, resulting in improved accuracy, recall, and F1-score. The model shows promising potential for phishing email detection with its high precision and effective combination of multiple algorithms. This suggests that the hybrid model with tuning has successfully enhanced the model's ability to make accurate predictions and effectively capture the positive instances. Therefore, the results support the use of the tuned hybrid model as this paper proposed.

The proposed hybrid model with TF-ID One of the major advantages of this hybrid model is improved accuracy. By combining multiple models through a voting classifier, the hybrid model can often achieve better accuracy compared to using a single model alone. This is because the voting classifier considers the predictions of multiple models, which can provide a more robust and diverse prediction compared to a single model.

However, the proposed hybrid model also has several disadvantages. One of the main disadvantages is the increased computational cost. The hybrid model can be more computationally intensive compared to using a single model, due to the need to train multiple models and make predictions from each. This can lead to longer training and testing times, which can be a hindrance for larger and more complex datasets.

Another disadvantage of the hybrid model is its complexity. The hybrid model is more complex to interpret compared to a single model, as it involves multiple models and requires understanding of how they interact with each other. This can make it more difficult to diagnose and debug issues with the model.

5. Conclusion

In conclusion, machine learning techniques have been proven to be effective in detecting phishing attacks. The results of the various models used in this study showed that different algorithms have their own strengths and weaknesses. Naive Bayes, K-Nearest Neighbors, Random Forest, Support Vector Machines, and Decision Trees are some of the commonly used algorithms in the field of phishing detection. The results of comparing the performance of the TF-IDF and Count Vectorization methods showed that the TF-IDF method had a higher accuracy and precision in most of the models. The proposed hybrid model, which combined Naive Bayes, Decision Tree, and K-Nearest Neighbors with parameter tuned and combined using a voting classifier, also showed promising results with an accuracy of 0.938, precision of 1.0, recall of 0.875, F1-score of 0.933, and ROC AUC score of 0.94. The hybrid model leverages the strengths of the individual models and combines them to make predictions. This can result in better performance compared to individual models, as well as reduced overfitting and improved generalization capabilities.

6. References

Adi Wijaya, & Achmad Bisri . (2016). Hybrid Decision Tree and Logistic Regression Classifier for Email Spam Detection. 2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE), Yogyakarta, Indonesia (p. 4). IEEE Xplore.

Akashsurya156. (2020). Kaggle. doi:https://www.kaggle.com/datasets/akashsurya156/phishing-paper1

Bhandari, A. (2023, 3 13). Understanding & Interpreting Confusion Matrices for Machine Learning (Updated 2023). doi:https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/

Doaa Mohammed Ablel-Rheem, Ashraf Osman Ibrahim, Shahreen Kasim, Abdulwahab Ali Almazroi, & Mohd Arfan Ismail. (2020). Hybrid Feature Selection and Ensemble Learning Method for Spam Email Classfication. 9, p. 8. International Journal of Advanced Trends in Computer Science and Engineering. doi:https://doi.org/10.30534/ijatcse/2020/3291.42020

Fang, Y., C. Z., C. H., L. L., & Y. Y. (2019). Phishing Email Detection Using Improved RCNN model with Multilevel vectors and attention mechanism. 7, 1-12.

F-score. (2021, March 12). Wikipedia. https://en.wikipedia.org/wiki/F-score

Fergus Toolan, & Joe Carthy. (2022). Feature Selection for Spam and Phishing Detection. 1-12.

Gallo, L., Maiello, A., Botta, A., & Ventre, G. (2021). 2 Years in the anti-phishing group of a large. Computers and Security, 1-18. doi:https://doi.org/10.1016/j.cose.2021.102259

Ganesan, K. (2019, December 5). 10+ Examples for Using CountVectorizer. Kavita Ganesan, PhD. https://kavita-ganesan.com/how-to-use-countvectorizer/

Harrison, O. (2018, September 10). *Machine Learning Basics with the K-Nearest Neighbors Algorithm*. Medium; Towards Data Science. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

IBM. (n.d.). What is Random Forest? | IBM. Www.ibm.com. https://www.ibm.com/topics/random forest#:~:text=Random%20forest%20is%20a%20commonly

J. Vijaya Chandra, Narasimham Challa, & Sai Kiran Pasupuletti. (2019, october). Machine Learning Framework To Analyze Against Spear Phishing. 8(12). doi:10.35940/ijitee.L3802.1081219

Jawale, D. S., Diksha S. Jawale , Kalyani R. Shinkar , & Kalyani R. Shinkar . (2018). Hybrid spam detection using machine learning. International Journal of Advance Research, Ideas and Innovations in Technology, 4(2), 1-6.

K, D. (2019, 6 14). Top 4 advantages and disadvantages of Support Vector Machine or SVM. Retrieved from https://dhirajkumarblog.medium.com/top-4-advantages-and-disadvantages-of-support-vector-machine-or-svm-a3c06a2b107

KARIM, A., S. A., (. I., B. S., & K. K. (2020). Efficient Clustering of Emails Into Spam and Ham:The Foundational Study of a Comprehensive. 8, 1 - 30.

Kearest Neighbors Algorithm. (n.d.). (IBM) Retrieved from https://www.ibm.com/my-en/topics/knn#:~:text=The%20k%2Dnearest%20neighbors%20algorithm%2C%20also%20known%20as%20KNN%20or,of%20an%20individual%20data%20point.

Kolmar, C. (2019, 10 19). 75 INCREDIBLE EMAIL STATISTICS [2023]: HOW MANY EMAILS ARE SENT PER DAY? Retrieved from https://www.zippia.com/advice/how-many-emails-are-sent-per-day/

Lew May Form, Kang Leng Chiew, San Nah Sze, & Wei King Tiong. (2022, 9 25). Phishing Email Detection Technique by using Hybrid Features. 5.

Liu, C. (2022, 9 20). More Performance Evaluation Metrics for Classification Problems You Should Know. Retrieved from https://www.kdnuggets.com/2020/04/performance-evaluation-metrics-classification.html

Narkhede, S. (2018, 6 27). Understanding AUC - ROC Curve. Retrieved from https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

NATHAN, L. (2021, 9 2). The Malaysian Reserve. Retrieved from themalaysianreserve: https://themalaysianreserve.com/2021/09/02/phishing-attacks-rising-since-pandemic-struck/

Narkhede, S. (2018, June 26). Understanding AUC - ROC Curve. Medium; Towards Data Science. https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

Phishing Email Collection. (n.d.). Www.kaggle.com. https://www.kaggle.com/datasets/akashsurya156/phishing-paper1

Phishing Email Data by Type. (n.d.). Www.kaggle.com. https://www.kaggle.com/datasets/charlottehall/phishing-email-data-by-type

Q1 2023 Phishing and Malware Report: Phishing Increases 102% QoQ. (n.d.). Www.vadesecure.com. https://www.vadesecure.com/en/blog/q1-2023-phishing-and-malware-report-phishing-increases-102-qoq#:~:text=In%20Q1%202023%2C%20Vade%20detected

RAZA , M., Jayasinghe, N. D., & Muslam, M. M. (2022). A Comprehensive Review on Email Spam Classification using Machine Learning Algorithms. 2021 International Conference on Information Networking (ICOIN), (pp. 1-6).

Sharma, S. (2021, 5 15). K-Nearest Neighbour: The Distance-Based Machine Learning Algorithm. Retrieved 5 15, 2021, from https://www.analyticsvidhya.com/blog/2021/05/knn-the-distance-based-machine-learning-algorithm/

Saini, A. (2021, August 29). *Decision Tree Algorithm - A Complete Guide*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/08/decision-tree-algorithm/#:~:text=A%20decision%20tree%20algorithm%20is

SVM Machine Learning Tutorial – What is the Support Vector Machine Algorithm, Explained with Code Examples. (2020, July 1). FreeCodeCamp.org. https://www.freecodecamp.org/news/svm-machine-learning-tutorial-what-is-the-support-vector-machine-algorithm-explained-with-code-examples/#:~:text=What%20is%20an%20SVM%3F

Thitithep Sitthiyot, & Kanyarat Holasut. ((112)2020, june 4). A simple method for measuring inequality. Retrieved from https://www.nature.com/articles/s41599-020-0484-6#:~:text=The%20Gini%20index%20is%20calculated,line%20(A%20%2B%20B).

Vazhayil, A., H. N., V. R., & S. K. (2018). Phishing Email Detection Using Classical Machine Learning Techniques. Proceedings of the 1st AntiPhishing Shared Pilot at 4th ACM International Workshop on Security and Privacy Analytics (IWSPA 2018), 2124, pp. 1-8. Arizona. doi:http://ceur-ws.org

Wikipedia Contributors. (2019, March 25). Accuracy and precision. Wikipedia; Wikimedia Foundation. https://en.wikipedia.org/wiki/Accuracy_and_precision

What is Naïve Bayes | IBM. (n.d.). Www.ibm.com. https://www.ibm.com/topics/naive-bayes#:~:text=The%20Na%C3%AFve%20Bayes%20classifier%20is

Worch, M. (2023, 9 23). Developing a Machine Learning Model to Identify Phishing Emails. Retrieved from https://ironscales.com/ironscales-engineering-corner-blog/developing-a-machine-learning-model-to-identify-phishing-emails/

Yuliya Kontsewaya, Evgeniy Antonov, & Alexey Artamonov. (2020). Evaluating the Effectiveness of Machine Learning Methods for. Procedia Computer Science 190 (2021) 479–486 (p. 8). Elsevier B.V. Retrieved from 10.1016/j.procs.2021.06.056.