ISSN 2409-2665
Journal of Logistics, Informatics and Service Science
Vol. 10 (2023) No.2, pp.249-261
DOI:10.33168/JLISS.2023.0217

Machine Learning Algorithms for Phishing Email Detection

Yoga Shri Murti and Palanichamy Naveen

Faculty of Computing and Informatics, Multimedia University,63100, Cyberjaya, Malaysia

p.naveen@mmu.edu.my

Abstract. Internet users are seriously endangered by phishing emails and keeping digital communication secure depends on their detection. The development of phishing strategies has necessitated continual research into increasingly sophisticated phishing email detection methods. Automatically spotting phishing emails has been demonstrated to be a powerful tool via machine learning (ML). In this study, we thoroughly examine current ML-based classifiers for accurately detecting phishing email. First, we employ a real-world dataset from Kaggle that has actual ratios of authentic and phishing emails. Then, Exploratory Data Analysis (EDA) is performed to understand the dataset better and identify obvious errors and outliers to help the detection process. Several ML methods, including Naive Bayes (NB), Support Vector Machine (SVM), Random Forest (RF), Decision Tree (DT), and K-Nearest Neighbors (KNN), are trained on the dataset. The receiver operating characteristic curve (ROC) and Area under the ROC curve (AUC) are the measures used to assess the models' performance. In addition, the time taken to train and test the model will be experimented to conclude the algorithm's efficiency in real-time. The findings show that random forest and decision tree classifier has the highest precision, recall, and f1-score, which concludes that these classifiers efficiently identify many positive instances. This study sheds important light on using ML for phishing email detection.

Keywords: Phishing email, detection, machine learning, classifiers

1. Introduction

In this modern era, most official communications use email. In addition, government and non – government organizations provide email services for people to get their service. Email is also an effective, faster, and, most importantly, cheaper way of communication. According to (Kolmar, 2019),333.2 billion emails are sent daily around the globe, which means the use of email is high, and the chances of getting unsolicited content like spam, fraud and phishing emails are also high.

Contrary to that, Malaysian reserve journal (Nathan, 2021), phishing emails attacking employees have escalated, according to 65% of Malaysians who have reported this pandemic. Phishing emails really can lead to many damages to people, businesses, companies, and other sectors that use email. Illegitimate emails can also result in significant harm besides being a nuisance and wasting one's time. It is because these emails often trick individuals into revealing sensitive information, such as passwords, credit card numbers, and other personal details, by posing as trustworthy entities through electronic communication.

Several studies conducted and documented on phishing email detection have used various methods, including effective email clustering, phishing email detection using traditional methods, and ML frameworks to analyze phishing. However, research on phishing email detection using ML has yet to produce a more accurate result and time-friendly detection; as a result, it is essential to enhance their performance in classifying phishing emails.

This study aims to address the challenge of phishing emails by finding the efficient and time-saving ML model to detect or classify phishing and legitimate emails. The approach is based on a comprehensive analysis and review of existing phishing email detection classifiers used in related studies.

In conclusion, ML has the potential to revolutionize the way and change the approach to detecting and preventing phishing attacks. ML can offer more precise and efficient phishing detection using large data sets and advanced algorithms than traditional methods. This can play a crucial role in safeguarding individuals and organizations from the negative consequences of phishing attacks.

The rest of the paper is organized as follows: related studies are discussed in Section 2. Section 3 discusses the material. The research methodology is addressed in Section 4, followed by the research setup and results in Section 5. Finally, in section 6, the study is concluded.

2. Related Studies

This project's literature review is a summary or review of journals and conference papers related to phishing email detection using the ML algorithm in section 2.1 and metrics that evaluates performance on detection in section 2.2. Following section 2.3 discusses the reviewed papers in overall.

2.1. Machine Learning

The authors of (Gallo et al., 2021) present a study that used supervised ML to analyze suspicious emails that may be phishing. The project had to overcome obstacles like the complexity of evaluating every email received, the requirement for moral users, the absence of security against single-victim attacks, and the limitations of supervised learning. The study relied on analysts manually identifying emails, which wasn't feasible for emails that weren't reported. Based on their precision, recall, and F-score, the authors evaluated the performance of several machines learning algorithms, including NB, Nearest Neighbour, Linear Support Vector Machine (Linear SVM), Radial Basis Function Support Vector Machine (RBF SVM), DT, RF, Adaptive Boosting (AdaBoost), and Multi-Layer Perceptron Neural Network (MLP Neural Net). The findings indicated that RF, which used 36 characteristics, could achieve a maximum of 95.2% Precision, 91.6% Recall, and 93.3% F-Score. According to the authors, future research might use unsupervised ML for the same objective.

Furthermore, the author of (Karim et al., 2020) proposed an anti-spam framework which evaluates based on the domain and email headers. They worked with six different clustering algorithms and concluded that their proposed approach, named 'Optics', resulted in an average of 3.5% better efficiency than spectral and k-means algorithms. Study (Gallo et al.,2021) designs and implements different classification and regression methods on email spam filter datasets where spear phishing methods are identified. Furthermore, ensemble methods like boosting, nagging, stacking and voting are added to existing algorithms for a higher classification result. From the evaluation process, they found that the KNN algorithm effectively implements and provides efficient prediction among the available algorithms.

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

Next, the Study (Vazhayil et al., 2018) focuses on developing phishing detection models through a non-sequential approach using classical ML classifications. First, the pre-processed data is converted by Term Document Matrix into 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 ML methods regarding classification accuracy, leading to high categorization rates. The authors also note that integrating data from an outside source can improve the suggested phishing detection architecture without relying on feature selection, necessitating 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 the bag of words (BOW) and body text features. The paper highlights the need to focus on different features, multi-algorithm systems, real-time spam classification, and small false positive rates. The authors found that multi-algorithm systems tend to perform better than a single algorithm, with algorithms such as NB and SVM being the most commonly used ML algorithms in this field.

Also, the paper (Yuliya et al., 2020) used a readily available dataset of 5728 emails from Kaggle, consisting 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. They divided the data into two sets: a training set (80%) and a testing set (20%). The authors employed f-measure, Accuracy, precision, recall, and the ROC area to assess the model quality. They used six 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%. The ML classifiers frequently used in related research are presented in Table 1.

Table 1. Summary of ML classifier used

No.	Authors	RF	DT	SVM	NB	LR	K-Means	KNN
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)			/	/			

2.2. Performance Metrics

Table 2 summarizes evaluation metrics highly used in related research papers: Accuracy, precision, recall, f-measures and confusion matric. AUC and ROC are the least used measures.

Table 2. 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)		/	/				

2.3. Discussion of reviewed papers

Some restrictions on using learning algorithms for phishing detection have indeed been found in the review of recent literature. The complexity of phishing emails, designed to look like legitimate emails, presents challenges for accurate classification. This paper aims to find efficient individual algorithms with high phishing detection accuracy. The 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 effective for spam classification compared to other machine learning methods. Most prior studies have used decision trees, Naive Bayes, SVM, and random forest for experiments. A few papers have used deep learning, such as Recurrent Convolutional Neural Networks (RCNN), as in the study (Fang et al., 2019) achieved high Accuracy. In prior studies, Naive Bayes, random forest, and decision trees are the most used and recommended machine learning methods, as they perform well and are efficient in time. The other limitations were that most researchers used a spam base dataset to conduct their projects because phishing emails need to be more concerned 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 a machine learning classifier to detect phishing emails, described in Section 3.1 – 3.6.

3.1. Data Obtained

The dataset used in this paper was obtained from Kaggle. The platform allows users to find, publish, and explore datasets in a web-based data science environment. The obtained dataset is a large

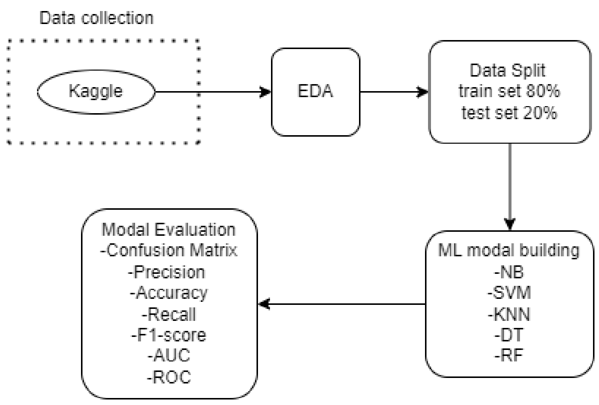

Fig. 1: Proposed Methodology

unbalanced, and pre-made dataset. The dataset was used to study the performance of ML models. algorithms with those previously used in the reviewed papers.

3.2. Description of Dataset

The dataset is a readily available phishing email dataset from the Kaggle platform. This dataset has 21 attributes and 451085 records. The records are numeric because tokenization is pre-applied in this dataset. All records are classed; phishing emails are labelled 0, whereas legitimate emails are labelled 1.

3.3. Exploratory Data Analysis (Eda)

EDA techniques are carried out for the dataset to have a look at data before continuing use for the next steps. EDA helps identify apparent errors, detect outliers, and better understand relations among the variables. Dataset 1 is unbalanced. EDA helps to drop Unwanted columns. The targeted columns were renamed. Then label encodes were used to encode the targeted column, such as Phishing emails labelled as 0 and legitimate emails labelled as 1. 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

3.4. Data Split

The dataset is divided into two parts, with 80% designated as the training set and 20% as the testing set, where 360868 records are for training and 90217 records for testing. This split is commonly used in machine learning because it uses a large enough sample to train the model and separate independent samples to evaluate its performance. This split also measures how well the model will generalize to new and unseen data using the captured relation patterns during training.

3.5. Machine Learning Algorithms

This paper implements the NB, SVM, RF, DT, and KNN machine learning classifiers. The models are discussed in Sections 3.5.1 to 3.5.5, respectively.

3.5.1 Naïve Bayes

The likelihood of every feature given to each class is calculated by Nave Bayes, who then multiplies these probabilities to obtain the possibility of every feature set given each class label. The forecast is then determined to be the class label with the highest likelihood. The NB algorithm assumes that, given the class label, all features are conditionally independent. Therefore, it is easier to calculate the likelihood of the class given the features, which is simply the product of the probabilities of each feature. NB regularly distributes gaussian features. Text classification and phishing screening are only two examples of the many issues that may be solved with the quick and straightforward NB algorithm. It is beneficial when many features and little data are available because it requires less training data to give good results. The dataset is cleaned using NB techniques before the texts are divided into tokens. Each token's text probability is determined. The likelihood of email is displayed in equation (1) for each word (A).

$$\mathrm{P(A)} = \frac{\frac{a\char`^spam}{s}}{w(\frac{a\char`^ham}{h} + \frac{a\char`^spam}{s})} \tag{1}$$

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 been determined.

3.5.2 Support Vector machine

Support Vector Machine (SVM) is a widely used machine learning technique that is applied to both classification and regression problems. It does this by creating a boundary, known as a hyperplane, that separates data into different groups and associates each group with a specific class. The hyperplane is defined by a line represented as in equation (2):

$$y = a^{*}x + b \tag{2}$$

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

They can be represented as a vector as shown in equation (3):

$$\mathrm{W^{*}\,X + b = 0} \tag{3}$$

W is the weight value, and b is a bias term. The study in this context uses a linear kernel SVM to find the best hyperplane to separate opinion data into two categories or binary classes. Finding the ideal hyperplane involves identifying the outermost data points in the two classes and considering them while determining the best hyperplane.

3.5.3 Random Forest

The ensemble learning family of machine learning algorithms includes the well-known Random Forest algorithm. Regression and classification issues are addressed by it. The main principle of RF is to create numerous decision trees during the training phase and then combine their predictions to make the final forecast. The final prediction will then be produced by combining the predictions made by each tree. RF can estimate the feature's importance, which can be useful for feature selection. Other than that, RF can be easily parallelized and suitable for large-scale problems. But, the drawback of RF is it is computationally expensive to train, as it requires to grow of multiple decision trees.

The Gini impurity index measures the impurity of a set of instances in terms of their class labels. The goal of using the Gini impurity index in decision tree-based algorithms, such as RF, is to determine the feature to use as the tree's root node that results in the lowest impurity or lowest Gini index. The lower the Gini index, the more homogeneous the class labels are in the set of instances. Mathematically, the Gini impurity index can be calculated by using equation (4).

$$\text{GINI INDEX} = 1 - \sum_{i=1}^{n}(Pi))^2 \qquad (4)$$
$$= 1 - [(P+)^2 + (P-)^2]$$

3.5.4 Decision Tree

A Decision Tree is a powerful and flexible algorithm widely used for various applications, including diagnosis, segmentation, and evaluation of phishing risks. It is a tree-structured algorithm that uses a series of rules to predict the value of a target variable. The decision tree consists of nodes, each representing a test on one of the input features. The branches of the node indicate the potential outcomes of the test, and the leaves of the tree represent the final predictions made by the model. The decision tree is constructed by repeatedly dividing the data into smaller subsets based on the features that impact the target variable most. This process equation (5) continues until the subset is homogeneous, meaning all instances have the same target variable value.

$$E(S) = \sum \bar{i} = 1 - \quad pi\ log2\ \text{pi} \qquad (5)$$

3.6.5 K-Nearest Neighbours

In KNN classification, a sample's prediction is based on the training data's KNN class labels. The number of neighbours that will be used to create the forecast depends on the value of the user-defined parameter k. The new sample's class label is then designated as the one used most frequently by its k closest neighbours. Regarding the underlying distribution of the data, KNN makes no assumptions. Additionally, simple to use, KNN requires time to locate the KNN for each new sample; therefore, the larger the dataset, the longer it takes to forecast the outcome. When used for classification, the KNN algorithm is based on the K instances most similar to a brand-new, unknown data point. To do this, a chosen distance metric, such as the Euclidean distance, is used to calculate the distance between the new data point and all other occurrences in the data collection. The program then classifies the new data point based on the majority of votes among the K examples that are the most similar. The Euclidean method, a standard distance measurement technique, is shown in equation (6).

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(yi - xi)^2} \qquad (6)$$

3.6. Performance Evaluation

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

Confusion Matrix: An algorithm's performance can be seen using a particular table structure called a confusion matrix, also known as an error matrix. This method is commonly used for supervised methods (in unsupervised learning, usually called a matching matrix). Each column of the matrix represents the examples in a predicted class, whereas each row represents the occurrences in an actual class. 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.

Table 3. Confusion Matrix

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

True positives (TP) are positive tuples that the classifier successfully categorized; TP represents the number of real positives. True negatives (TN) are the negative tuples the classifier correctly categorized; the total number of true 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 incorrectly classified as negatives. The number of false negatives, FN, shall be used.

Accuracy: Accuracy is a performance measure that explains how well a model performs in classifying instances correctly. It is calculated as the ratio of the number of correct predictions made by the model to the total number of predictions made. This metric indicates the model's performance in all classes, not just positive or negative cases.

Precision: Precision is a metric that measures Accuracy in making accurate predictions, especially for the minority class. Of all the positive predictions, it calculates the percentage of correct, true positives produced by the model. Precision is essential when working with imbalanced datasets, where the minority class is often more interesting or important to study.

Recall: Recall is a metric that measures the Accuracy of positive predictions in a model. It calculates the number of correct positive predictions made by the model out of all the positive cases. Recall gives an idea of the instances where the model failed to identify positive cases correctly.

F1-score: The harmonic mean of recall and precision is used to generate the evaluation metric known as the F1-score. It is a useful alternative to using precision and recall separately, as it provides a balance between the two measures.

ROC curve: The Accuracy of a binary classifier is visually represented by the Receiver Operating Characteristic (ROC) curve. This technique shows 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.

AUC: The AUC is a metric of how effectively a binary classifier can differentiate between two classes (AUC). It provides a clear explanation 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 (13). 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.

4. Evaluation and Discussion

This section displays the results achieved by the proposed approach. Section 4.1 discusses the time taken to test and train, along with the Accuracy and precision of the algorithms. Section 4.2 analyses all the machine learning classifiers used in this study. Section 4.3 shows the graphical results of the overall performance.

4.1. Time taken to Train and Test

Table 4 illustrates the time taken to train and test along with the train and test model's accuracy of ML classifiers. Based on table 4, all the models perform relatively well in terms of Accuracy. However, the performance in terms of time taken to train and test varies significantly between the models. The NB model is the fastest to train and test, taking only 0.2 seconds to train and 0.1 seconds to test. It achieves a train accuracy of 96.754% and a test accuracy of 96.669%.

Table 4. Time taken to train and test & Accuracy

Classifier	Train		Test	
	Time	accuracy	Time	accuracy
NB	0.2 s	96.754	0.1 s	96.669
KNN	0.5 s	98.576	33.9s	98.316
RF	50.4 s	99.999	1.1 s	99.451

SVM	3m 59.5s	98.187	2m 43.9s	98.130
DT	2.9 s	100.0	0.0s	99.104

The KNN model takes longer to train and test, with a training time of 0.5 seconds and a testing time of 33.9 seconds. However, it achieves a higher train accuracy of 98.576% and a test accuracy of 98.316%. The RF model has the longest training time, taking 50.4 seconds. However, it only takes 1.1 seconds to test and achieves the highest train accuracy of 99.999% and test accuracy of 99.451%.

The SVM model takes significantly longer to train, with a training time of 3 minutes and 59.5 seconds and a testing time of 2 minutes and 43.9 seconds. It achieves a train accuracy of 98.187% and a test accuracy of 98.130%. The DT model is the fastest to train, taking only 2.9 seconds. It achieves a perfect train accuracy of 100% but a slightly lower test accuracy of 99.104%.

The RF model performs the best in Accuracy but has the longest training time. On the other hand, the NB model is the fastest but has slightly lower Accuracy than some of the other models. Therefore, the choice of model will depend on the application's specific requirements, such as the desired level of Accuracy and the adequate time for training and testing.

4.2. Confusion Matrix Results

The confusion matrix of the ML classification algorithms are shown in Table 5. NB: There have been 86358 cases of phishing emails that were predicted as TP, 854 cases of legitimate emails that were predicted as TN, 833 cases of phishing emails that were misidentified as FP, and 2172 cases of non-phishing emails that were phishing emails FN.

SVM: There have been 88529 cases of phishing emails that were predicted TP, 1 case of legitimate emails that were predicted as TN, 1686 cases of phishing emails that were misidentified as FP, and 1 case of non-phishing emails that were phishing emails FN.

RF: There have been 88489 cases of phishing emails that were predicted as TP, 1232 cases of legitimate emails predicted as such TN, 455 cases of phishing emails misidentified as FP, and 41 cases of non-phishing emails that were phishing emails FN.

DT: There have been 88160 cases of phishing emails predicted as TP, 1239 cases of legitimate emails predicted as such TN, 448 cases of phishing emails misidentified as FP, and 370 cases of non-phishing emails that were phishing emails (FN).

KNN: There have been 88401 cases of phishing emails predicted as such TP, 297 cases of legitimate emails that as such TN, 1390 cases of phishing emails misidentified as such FP, and 129 cases of non-phishing emails that were phishing emails (FN).

Table 5. Confusion Matrix of ML classifiers

Classifiers		Phishing email	Non-Phishing email
NB	Phishing email	86358	2172
	Non -phishing email	833	854
SVM	Phishing email	88529	1
	Non -phishing email	1686	1
RF	Phishing email	88489	41
	Non -phishing email	455	1232
DT	Phishing email	88160	370
	Non -phishing email	448	1239
KNN	Phishing email	88401	129
	Non -phishing email	1390	297

Table 6 shows the rate of correctly predicted and incorrectly predicted results. All of the machine learning models have relatively high accuracy rates, ranging from 96.669% for NB to 99.45% for RF.

It indicates that the models can accurately classify the majority of instances in the dataset. On the other hand, in terms of misclassification rates (Email Phishing, Vishing & Other Types of Attacks, 2023) all of 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 6. 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

Table 7. shows RF has high results, which indicate that it is very suitable for phishing email detection. Moreover, relatively high precision from DT classifiers supports making accurate prediction and a moderate number of positive instances. KNN is also capable of making accurate predictions but cannot identify positive cases. Next, the classifiers NB and SVM could be more efficient in identifying positive prediction compared to other examined classifiers. Fig. 2 shows the Accuracy and precision of machine learning classification algorithms.

Table 7. Precision, Recall & F1-score

Measures / Classifiers	Precision	Recall	F1-score
NB	28.22	50.62	36.24
SVM	50.0	0.059	0.0118
RF	96.78	73.03	83.24
DT	77.44	73.44	75.18
KNN	69.72	17.61	28.11

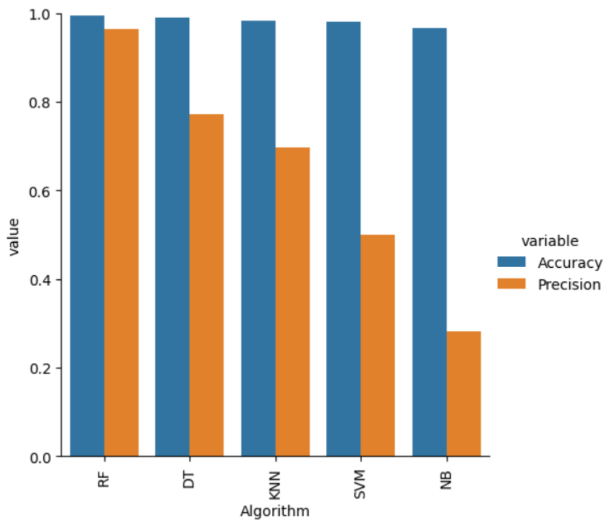

Fig.2: accuracy and precision of machine learning classification algorithms

ROC: It provides a graphical visualization of the classifier's performance. Fig.3. shows the ROC curve and area of DT, RF, NB, KNN and SVM. As a result of overall findings based on Fig. 3, the RF classifier and DT have high ROC area and outperform the other three classifiers. Based on the graph, it is seen that DT predicted the phishing emails efficiently, whereby the area of ROC curve is 0.87. Meanwhile, RF's ROC area is 0.86. However, SVM performs poorly with ROC curve area 0.5 due to the algorithm's limitation. NB has the moderate performance of ROC area 0.74 and KNN ROC area is 0.59.

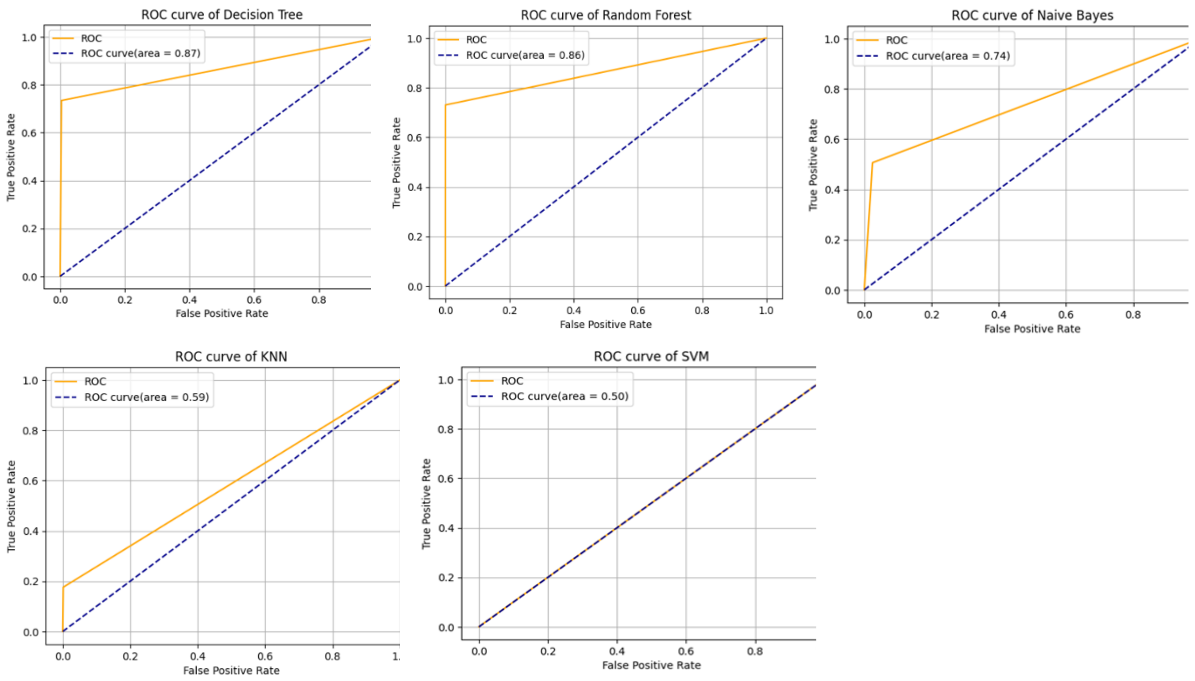

Fig. 3: The ROC curve of DT, RF, NB, KNN and SVM

4.3. Overall Findings

As overall finding, RF has the highest accuracy rate and precision/recall/F1-score measures, while NB has the lowest misclassification rate. If the goal is to maximize recall, then NBs may be a good choice. If the goal is to optimise precision, then RF may be a good choice. Finally, if the goal is to balance both measures, then DT may be a good choice. However, it's important to note that these measures are based on a specific dataset and may vary depending on the characteristics of other datasets. Besides, SVM's performance could be better when it comes to a large dataset because it is highly dependent on the size of the dataset and it is significantly time consuming to train the model. On the other hand, KNN is a moderate classifier because it produces better Accuracy while taking a reasonable amount of time to train and test.

5. Conclusion

In conclusion, it has been demonstrated that machine learning approaches successfully spot phishing emails. The outcomes of the multiple classifiers employed in this study demonstrated the diversity in each machine learning algorithm. Different machine learning algorithms have different strengths and weaknesses in detecting phishing emails, with RF having the highest accuracy rate and NB having the lowest misclassification rate. DT may be an excellent choice to balance both measures. At the same time, SVM's performance is limited for large datasets, and KNN is a moderate classifier that produces better Accuracy in a reasonable amount of time. Algorithms such as RF, DT, and KNN are suitable for detecting phishing emails in real time. However, the choice of which algorithm to use depends on the specific requirements of the problem, dataset size, and available computational resources. Overall, all machine learning models effectively classify or detect phishing emails.
Our future work will focus on finding efficient feature selection methods to minimize the irrelevant and

redundant attributes in the detection process. Furthermore, feature selection methods help to increase the power of the machine learning algorithm to achieve more accurate predictions in the phishing email detection process.

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.

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.

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

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.

K-Nearest 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/

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/

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

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.