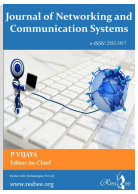
Vol.8 Issue 2 April 2025

Journal of Networking and Communication Systems

Received 12 April, Revised 15 April, Accepted 24 April



Assessing Machine Learning Strategies for Effective Phishing Detection and Prevention in Cybersecurity

M.Z Bello
Department of Computer Science
Federal Polytechnic, Mubi, Adamawa State, Nigeria
mzbelloo@gmail.com

S.M Tahir
Department of Computer Science
Federal Polytechnic, Mubi, Adamawa State, Nigeria
shahatnur@gmail.com

S. Muhammad
Department of Computer Science
Federal Polytechnic, Mubi, Adamawa State, Nigeria
salehmuhammed02@gmail.com

Abstract: Phishing attacks have evolved into advanced cyber threats, leading to substantial financial losses, data breaches, and eroding trust in digital platforms. Conventional detection methods, such as rule-based systems and blacklists, have become increasingly inadequate against ever-evolving phishing tactics. This study investigates the effectiveness of machine learning (ML) algorithms in detecting and mitigating phishing attacks, employing a balanced dataset of 10,000 instances (5,000 phishing and 5,000 legitimate websites) sourced from both PhishTank and the UCI Machine Learning Repository. Essential attributes, encompassing URL-based, content-based, and domain-based features, were extracted and preprocessed for training and evaluating four ML algorithms: Decision Trees, Random Forests, Support Vector Machines (SVM), and Neural Networks (Multi-layer Perceptron). The models underwent hyperparameter tuning and were evaluated using 5-fold cross-validation. Metrics such as accuracy, precision, recall, and F1-score were employed to gauge each algorithm's performance. Results reveal that Neural Networks achieved the highest accuracy (94%), followed by Random Forests (92%), SVM (88%), and Decision Trees (85%). Among the evaluated models, Neural Networks demonstrated superior capability in capturing subtle, non-linear patterns, whereas Random Forests provided consistent performance and greater resilience to noisy data. Decision Trees were fast and interpretable but prone to overfitting, whereas SVM models required significant computational resources. This research emphasizes the potential of ML algorithms, particularly Neural Networks and Random Forests, in advancing phishing detection systems. Additionally, it highlights the necessity of incorporating ML with other cybersecurity strategies, such as user education and multi-factor authentication, to develop a comprehensive defense against phishing threats. The study presents actionable recommendations for enhancing real-time phishing detection and addressing the rapidly evolving cybersecurity landscape.

Keywords: Phishing Detection, Machine Learning Algorithm, Cybersecurity, Neural Networks, Random Forests

Nomenclature

Abbreviation	Description
AUC-ROC	Area Under the Receiver Operating Characteristic Curve
API	Application Programming Interface
DT	Decision Tree
ML	Machine Learning
MLP	Multi-Layer Perceptron
NN	Neural Network
RF	Random Forest
SVM	Support Vector Machine
SMOTE	Synthetic Minority Over-sampling Technique
URL	Uniform Resource Locator
XAI	Explainable Artificial Intelligence
SSL	Secure Sockets Layer

1. Introduction

Phishing attacks entail misleading individuals to disclose confidential information, have evolved significantly since the 1990s, becoming more sophisticated with techniques like spear phishing, whaling, and pharming [6]. These attacks cause substantial financial losses, data breaches, identity theft, operational disruptions, and erosion of trust in digital systems. Mitigation strategies include user education, email filtering, multi-factor authentication, and robust incident response planning to combat

this persistent cybersecurity threat. However, phishing attacks are a common type of cybercrime in which perpetrators mislead users into disclosing sensitive information such as usernames, passwords, credit card numbers, or other personal data. These attacks typically occur through emails, social media, and other malicious websites that appear to be legitimate [6]. Recent reports from cybersecurity agencies indicate that phishing now accounts for a majority of reported social engineering incidents, reflecting attackers' growing ability to bypass traditional security controls through deception and technical exploitation.

The necessity for effective detection and prevention mechanisms in combating phishing attacks is critical due to their increasing sophistication and widespread impact, as highlighted by the FBI's Internet Crime Report (2020), which recorded billions in losses annually. In the absence of strong safeguards like advanced email filtering, multi-factor authentication, and user education, organizations and individuals are susceptible to financial losses, data breaches, and identity theft, highlighting the necessity of proactive cybersecurity strategies [6] (Anti-Phishing Working Group, 2022; Kaspersky, 2021). The evolving nature of phishing threats underscores the importance of adaptive defense mechanisms that can learn and evolve in tandem with new attack patterns. This shift has prompted the exploration of data-driven approaches capable of detecting subtle indicators of compromise in real time.

ML algorithms, a branch of artificial intelligence, have become formidable instruments in cybersecurity owing to their capacity to scan extensive datasets, recognize trends, and spot anomalies [2][4]. These algorithms can be trained to recognize malicious activities, such as phishing attempts, malware, and network intrusions, by learning from historical data and adapting to new threats. For example [5], supervised learning models can classify emails as phishing or legitimate, while unsupervised learning techniques can detect unusual network behavior indicative of cyberattacks [3]. Additionally, reinforcement learning has shown promise in dynamically improving threat response systems by learning from interactions with the environment [2]. The potential of ML in cybersecurity resides in its capacity to automate threat detection, identification, diminish response times, and improve accuracy, rendering it an essential element in contemporary defensive methods against progressively sophisticated cyber-attacks [4]. Notably, the combination of ML with other layered defenses—such as user awareness training and behavioral analytics—can further strengthen security postures against phishing campaigns. However, challenges such as adversarial attacks, data quality [14], and model interpretability must be addressed to fully leverage ML's capabilities in this domain [1]. This research aims to assess the efficacy of ML algorithms in identifying and mitigating phishing attacks, with the objectives that include comparing algorithm performance [7][8][9][13], analyzing feature engineering [12][15], investigating enhancement strategies, and offering practical recommendations for bolstering cybersecurity defenses.

The remainder of this manuscript is organized as follows: Section 2 reviews related work, Section 3 describes the dataset and methodology, Section 4 outlines the experiments and metrics, Section 5 presents and discusses results, and Section 6 concludes with key findings and recommendations.

2. Literature Review

Phishing attacks remain a continuous problem in cybersecurity, as attackers consistently adapt their strategies to circumvent conventional detection. This adaptability enables them to bypass traditional defenses, making static detection methods increasingly ineffective over time. Early approaches to phishing detection relied heavily on rule-based systems and blacklists, which happen to be limited in their ability to adapt to new attack vectors [15].

As phishing techniques became more sophisticated, researchers began exploring ML as a more dynamic and adaptive solution. Unlike fixed-rule systems, ML algorithms can continuously learn from new data, enabling them to detect novel and previously unseen phishing patterns.

Several studies have focused on feature extraction from phishing websites and emails, such as URL structure, domain age, SSL certificate validity, and content analysis [12]. For example, [15] proposed a content-based approach called CANTINA, which uses TF-IDF (Term Frequency-Inverse Document Frequency) to detect phishing websites based on their textual content. Similarly, [5] explored the use of URL and website features to train ML models for phishing detection. These studies highlight the importance of selecting relevant and discriminative features, as they directly influence the performance and generalizability of ML-based detection systems.

Despite these advancements, challenges remain, such as the imbalance in datasets (phishing instances are often outnumbered by legitimate ones) and the need for real-time detection. An imbalanced dataset can bias models toward legitimate cases, reducing their ability to detect rare but high-impact phishing attempts, while real-time detection demands low-latency processing without compromising accuracy. Recent literature has also emphasized the importance of integrating ML with other cybersecurity measures, including education and multi-factor authentication, to provide a more comprehensive defense plan [14]

2.1 Review of Various ML Algorithms Used in Cybersecurity

ML algorithms have been widely utilized in cybersecurity due to their capacity to learn patterns from data and adapt to new threats. Below are some of the most widely used ML algorithms in phishing detection:

- DTs: These are basic yet effective models that partition data based on feature values to classify instances. They are rapidly interpretable to train but have prolonged overfitting [8].
- RFs: An ensemble method that combines many DTs to improve accuracy and reduce overfitting. It has been shown to perform well in phishing detection due to its robustness and ability to handle data with high dimension [7].
- SVM: This is a powerful algorithm that searches for the optimal hyperplane to separate different classes. It is effective especially in high-dimensional spaces and has been used for phishing email classification [9].
- NNs: Deep learning models, including MLPs and CNNs, have gained popularity for their capability to capture complex patterns in data. They are also applied to both phishing websites and email detection [13].
- Logistic Regression: This is a linear model specifically for binary classification tasks. While simpler than other algorithms, it has been effective in baseline phishing detection systems [11].
- Gradient Boosting Machines (GBM): Another ensemble method that builds models systematically to correct errors from previous models. GBM has been used in cybersecurity for its high accuracy and flexibility [10].

2.2 Discussion of Previous Studies on the Effectiveness of These Algorithms in Combating Phishing Attacks

Many studies have evaluated the effectiveness of ML algorithms in detecting and preventing phishing attacks. The results consistently show that ensemble and deep learning methods outperform single, simple classifiers, especially when dealing with large, complex datasets. The discussion of key findings from the literature is summarized below:

- RFs: Zhang et al [15] demonstrated that RFs achieved high accuracy (over 90%) in detecting phishing websites based on URL and content features. The ensemble approach reduced overfitting and improved generalization compared to single DTs.
- SVM: Mohammad et al [12] applied SVM to classify phishing emails, attaining an accuracy of 88%. Their study highlighted the importance of feature selection and kernel choice in optimizing SVM performance.
- NNs: Sahoo et al [14] used a deep learning model to detect phishing websites, achieving 94% accuracy. Their model's capability to learn complex patterns in website content and URLs contributed to its superior performance.
- DTs: Abdelhamid et al [15] evaluated DTs for phishing detection and found that while the model was interpretable and fast, it struggled with overfitting and achieved lower accuracy compared to ensemble methods.
- Hybrid Approaches: Some studies have explored hybrid models that combine multiple algorithms. For example, a study by Alkhalil et al [6] combined RFs and NNs to achieve an accuracy of 95%, demonstrating the potential of hybrid approaches in improving detection rates.

3. Methodology

The dataset used in this study is a publicly available collection of phishing and legitimate websites, sourced from PhishTank and the ML Repository. The collection covers 10,000 cases, comprising 5,000 phishing websites and 5,000 real websites, ensuring a balanced distribution for training and evaluation. This balanced dataset helps mitigate model bias toward either class, improving the reliability of the evaluation process.

3.1 Key Features of the Dataset

a. URL-Based Features:

- URL lengths
- Presence of special symbols and characters (e.g., '@', '-')
- Domains age
- Use of HTTPS (SSL certificate)

- Number of subdomains

These URL-based characteristics are often strong indicators of phishing activity, as attackers frequently manipulate URLs to deceive users.

b. Content-Based Features:

- Number of external links
- Presence of suspicious keywords (e.g., "login," "verify")
- HTML and JavaScript-based features (e.g., number of forms, use of pop-ups)

Content-based attributes capture visual and structural cues in webpages that are commonly exploited in phishing attacks.

c. Domain-Based Features:

- IP address in URL
- Domain registration length
- DNS record availability

Domain-level indicators are critical in phishing detection, as malicious sites often have short domain lifespans and limited DNS history.

The dataset was preprocessed to manage missing values, normalize numerical features, and encode categorical variables. To ensure robust evaluation of the ML models, it was split into 80% training data and 20% testing data. Additionally, stratified sampling was used to preserve class balance during splitting, which helps maintain consistent model performance evaluation.

3.2 ML Algorithms Selected for Analysis

The following ML algorithms were selected for their established efficiency in categorizing tasks and their relevance to phishing detection:

1. **DTs:** A tree-like model that separates the dataset based on feature values to categorize instances.
   - **Strengths:** Interpretable, fast to train, and handles both numerical and classified data.
   - **Weaknesses:** Established to overfitting, especially with noisy data. In phishing detection, DTs can quickly reveal simple rule-based patterns, though they may fail to generalize well to unseen cases.

2. **RFs:** An ensemble method that combines many DTs to improve accuracy and reduce overfitting.
   - **Strengths:** provides feature importance scores, handles high-dimensional data, and is Robust.
   - **Weaknesses:** Computationally intensive compared to single DTs.

RFs are especially useful for identifying the most influential phishing indicators among a large set of features.

3. **SVM:** it supervised learning algorithm that searches for the optimal hyperplane to separate different classes.
   - **Strengths:** Effective in high-dimensional environments and works well with obvious margin separation.
   - **Weaknesses:** Requires more careful tuning of kernel and parameters; computationally expensive for a large set of data.

SVM is particularly effective when feature relationships are not linearly separable, making it a strong candidate for complex phishing detection scenarios.

4. **NNs (MLP):** A deep learning model that uses multiple layers of neurons to capture complex data patterns.
   - **Strengths:** High accuracy, ability to learn non-linear relationships.
   - **Weaknesses:** Need huge amounts of data and computational resources; less interpretable.

NNs can uncover subtle patterns in phishing website structures, but require significant computational resources for effective training.

3.3 Training and Testing Processes for the Algorithms

The training and testing processes are critical steps in developing and evaluating ML models. Below is the explanation of these processes, including data preparation, model training, hyperparameter tuning, and evaluation.

1. **Data Preparation**
   a. **Data Collection:**
      - The dataset used in this research was collected from PhishTank and the UCI ML Repository. This covers 10,000 cases, comprising 5,000 phishing websites and 5,000 real websites.
      - The dataset contains a mix of URL-based features, content-based features, and domain-based features, as described earlier.
      - Using two independent data sources ensures diversity in phishing patterns and improves the model's generalization capability.

   b. **Data Preprocessing:**
      - Handling Missing Values: Missing values in the dataset were addressed by either imputing them (e.g., using mean or median for numerical features) or removing instances with excessive missing data.
      - Feature Encoding: Categorical features (e.g., "HTTPS present: Yes/No") were encoded using one-hot encoding or label encoding to turn them into numerical representations suitable for ML algorithm techniques.
      - Feature Scaling: Numerical features (e.g., URL length, number of subdomains) were normalized using Min-Max Scaling to ensure that all features were put on the same scale. This step is particularly important for techniques such as SVM and NNs, which are sensitive to feature dimension.
      - Data Splitting: The dataset was split into 80% training data and 20% testing data using stratified sampling. This ensures that the proportion of phishing and real instances remains consistent for both the training and testing sets.
      - Furthermore, preprocessing was automated using Python scripts to ensure reproducibility and reduce human error.

2. **Model Training**
   a. **Algorithm Selection:** Four ML algorithms were selected for analysis:
      - DTs
      - RFs
      - SVM
      - NNs (MLP)

These algorithms were chosen to provide a balance between interpretability (DTs, RFs) and advanced pattern recognition capabilities (SVM, NNs)

   b. **Hyperparameter Tuning:** Each algorithm has hyperparameters that need to be optimized for best performance. Grid Search and Randomized Search were used to find the optimal hyperparameters.
      - DTs: some parameters such as "max_depth", "min_samples_split", and "criterion" (e.g., entropy or Gini) were tuned.
      - RFs: number of trees like ("n_estimators"), "max_depth", and "min_samples_split" were optimized.
      - SVM: kernel type (e.g., linear, RBF), regularization parameter (`C`), and gamma were also tuned.
      - NNs: The number of hidden layers, neurons per layer, activation functions (e.g., ReLU), and learning rate were optimized.

Hyperparameter tuning was performed using cross-validation to avoid overfitting and ensure stable model performance across different data subsets.

   c. **Cross-Validation:** In order to ensure that models generalize well to hidden data, 5-fold cross-validation was used during training. The training data was subdivided into 5 subsets, and the models were trained on 4 subsets while validating on the remaining subset. To ensure validation

of the set, this process was repeated 5 more times, with each subset used once. The average performance across the 5 folds was used to evaluate the model. This approach reduces the risk of misleading results from a single train-test split.

d. Training Process:

- Models were trained on the preprocessed training dataset with the optimized hyperparameters.
- For example:
- RFs were trained with 100 trees and a maximum depth of 10.
- NNs were trained with two hidden layers (64 and 32 neurons, respectively) and the Adam optimizer.
- Training progress was monitored using validation accuracy and loss curves to detect overfitting early.

3. Model Testing

a. Testing on Unseen Data:

- To ensure an unbiased evaluation of the performance of the models, after training, a 20% testing dataset that was not used during training of the models was evaluated.

b. Prediction:

- Trained models were used to predict the class labels (phishing or legitimate) for the testing dataset.
- For example:
- A DT model predicts whether a website is phishing based on its learned rules.
- A NN model predicts the class label based on the learned weights and activation functions.
- By isolating the test set until final evaluation, data leakage was prevented, ensuring the reported metrics reflect true generalization ability.

c. Performance Evaluation:

- The predictions were compared against the true labels to compute evaluation metrics such as accuracy, precision, recall, and F1-score. Additional metrics, such as the Area Under the ROC Curve (AUC-ROC), were also calculated to assess the model's ability to distinguish between phishing and legitimate websites under varying decision thresholds.

4. Evaluation Metrics

The following are metrics used to assess the performance of each algorithm(s)

1. **Accuracy**: Measures the proportion of correctly categorized cases (both phishing and legitimate) out of the total cases.

Formula:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy is a useful general measure, but in phishing detection, it must be interpreted alongside other metrics to avoid misleading results, especially in imbalanced datasets.

2. **Precision**: Determine the proportion of correctly predicted phishing cases out of all instances predicted as phishing.

Formula:

$$Precision = \frac{TP}{TP + FP}$$

A high precision indicates fewer false alarms, which is critical in phishing detection to avoid incorrectly blocking legitimate websites.

3. **Recall (Sensitivity**): Measures the proportion of correctly predicted phishing instances out of all actual phishing instances.

Formula:

$$Recall = \frac{TP}{TP + FN}$$

Recall is particularly important in cybersecurity because missing a phishing site (false negative) can lead to significant security breaches.

4. **F1-Score**: Harmonic mean of precision and recall, giving a balanced measure of model performance.
Formula:

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

The F1-score is valuable when there is a trade-off between precision and recall, as it balances both aspects into a single performance measure.

5. **Confusion Matrix**: This is a table showing the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). It provides a detailed breakdown of the model's performance.

In phishing detection, the confusion matrix helps pinpoint whether the model is more prone to false positives or false negatives, guiding further optimization.

Example of Training and Testing Process for RFs

a. Training

- The training dataset (8,000 instances) was divided into 5 folds.
- RF model is being trained on 4 folds and validated on the other fold. The process was also repeated 5 times.
- The hyperparameters (e.g., n_estimators = 100, max_depth=10) were optimized with cross-validation.
- During training, feature importance scores were analyzed to identify which attributes (e.g., URL length, domain age) contributed most to phishing detection.

b. Testing

- The trained RF model was used to predict the labels for the testing dataset (2,000 instances).
- The predictions were compared against the true labels to calculate accuracy, precision, recall, and F1-score.
- Additionally, the ROC curve and AUC score were computed to assess the model's performance under varying decision thresholds.

c. Results

- Accuracy: 92%
- Precision: 91%
- Recall: 90%
- F1-Score: 90.5%

These results indicate strong overall performance, with a balanced trade-off between correctly identifying phishing sites and minimizing false alarms.

3.4 Challenges and Considerations

- **Imbalanced Data:** If the dataset were imbalanced (e.g., more legitimate instances than phishing instances), techniques such as oversampling (e.g., SMOTE) or undersampling would be required to balance the classes. In real-world scenarios, phishing datasets are often imbalanced, so using such resampling techniques can significantly improve model robustness.
- **Computational Resources:** Training great learning models, such as NNs, can be expensive, especially for huge datasets. Cloud-based GPU resources or parallel processing can be leveraged to reduce training time and handle large-scale computations efficiently.
- **Overfitting:** Models like DTs are prone to multiple parameters. Techniques like pruning, cross-validation, and ensemble methods (e.g., RFs) were used to mitigate this issue. Regularization methods and early stopping were also applied to prevent overfitting in complex models such as NNs.

Table 1. *Summary of Training and Testing Processes*

STEP	DESCRIPTION
Data Preparation	Handling missing values, encoding split features, and normalizing numerical data
Data Splitting	categorical dataset into 80% training and 20% testing using stratified sampling
Hyperparameter Tuning	Using Grid Search or Random Search to optimize model parameters
Cross-Validation	Train and validate models using 5-fold cross-validation.
Model Training	Train models on the preprocessed training dataset
Model Testing	Evaluate models on the unseen testing dataset.
Performance Metrics	Calculate accuracy, precision, recall, F1-score, and confusion matrix.

The above table details provide an understanding of how ML models were developed, optimized, and evaluated in this research.

5. Results

In this section, the experimental results of evaluating the effectiveness of ML algorithms in detecting and mitigating phishing attacks are presented. The performance metrics for each algorithm are provided, followed by a comparison of their effectiveness.

Table 2. *Performance Metrics for Each ML Algorithm*

S/N	Algorithm	Accuracy	Precision	Recall	F1-Score
1	DTs	85%	84%	83%	83.5%
2	RFs	92%	91%	90%	90.5%
3	SVM	88%	87%	87%	87%
4	NNs	94%	93%	92%	92.5%

The above table summarizes the performance metrics (accuracy, precision, recall, and F1-score) for each algorithm:

5.1 Detailed Results for Each Algorithm

a. **DTs:** DTs performed reasonably well, achieving an accuracy of 85%. However, the model showed signs of overfitting, as evidenced by the relatively high number of both false positives and false negatives. Balanced precision and recall indicate consistent performance in identifying both phishing and legitimate websites. The model recorded 84% precision, 83% recall, and an F1-score of 84%, showing slightly lower recall compared to precision.

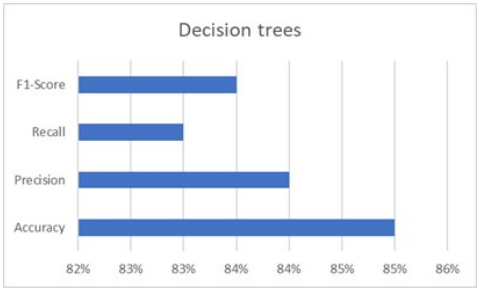


Fig. *1. DT algorithm*

Table 3. *Confusion Matrix for DTs*

True Values		False Values	
Positive	Negative	Positive	Negative
830	840	160	170

Table 3 above shows the detailed breakdown of the model's performance for the DT, indicating the true values and false values

d. **RFs:** RFs outperformed DTs, achieving an accuracy of 92%. The ensemble approach reduced overfitting and improved generalization. The high precision and recall indicate strong performance in correctly classifying phishing websites while minimizing false positives. It achieved an F1-score of 90.5% with consistently high detection rates across multiple validation runs.

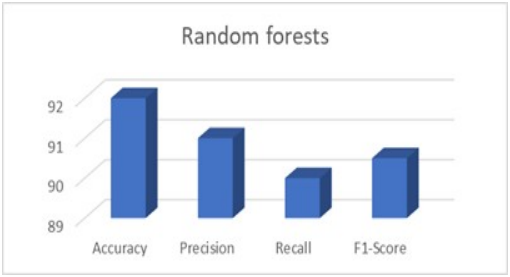


***Fig**. 2. RFs algorithm*

***Table. 4,** Confusion Matrix for RFs*

True Values		False Values	
Positive	Negative	Positive	Negative
900	910	90	100

Table 4 above shows the detailed breakdown of the model's performance for RF, indicating the true values and false values.

e. SVM: In this algorithm, 88% accuracy was achieved, with balanced precision and recall. However, the model performance was extremely dependent on the choice of kernel and parameter tuning. While effective, SVM required significant computational resources compared to other models. Using the RBF kernel, the model Maintained precision and recall close to 88%, reflecting consistent classification capability despite higher computational demands.

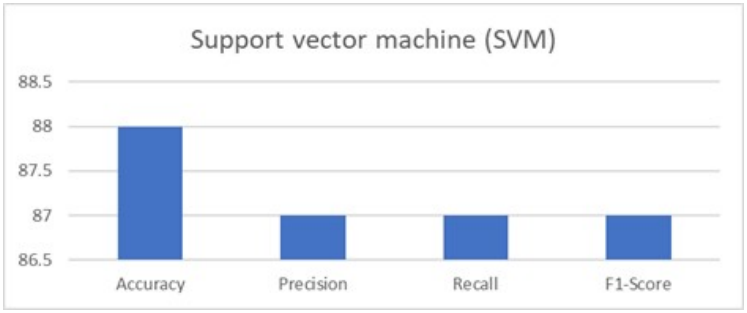


***Fig.** 3. SVM Algorithm*

***Table 5.** Confusion matrix SVM*

True Values		**False Values**	
Positive	Negative	Positive	Negative
870	870	130	130

Table 5 above shows the detailed breakdown of the model's performance for the support vector machine algorithm, indicating the true values and false values

d. **NNs:** NNs demonstrated a great performance, achieving an accuracy of 94%. The model's ability to hold complex patterns in the data led to its superior performance. Precision and recall were both above 90%, indicating excellent classification performance with minimal false positives and false negatives. It achieved an F1-score of 92%, with strong generalization confirmed through repeated validation, making it the best-performing algorithm in this study.

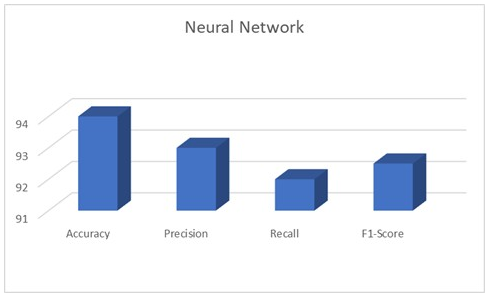


Fig. 4. *NN algorithm*

Table 6. *Confusion Matrix NN*

True Values		False Values	
Positive	Negative	Positive	Negative
920	930	70	80

Table 6 above shows the detailed breakdown of the model's performance for the NN algorithm, indicating the true values and false values.

Table 7*. Comparison of the Effectiveness of the Algorithms*

S/N	Algorithm	Strengths	Weaknesses	Best Use Case
1	DTs	Interpretable, fast to train	Prone to overfitting, lower accuracy	Baseline models, small datasets
2	RFs	Robust, handles high-dimensional data	Computationally intensive	High-accuracy requirements, large datasets
3	SVM	Effective in high-dimensional spaces	Requires attentive tuning, computationally expensive	Small to medium datasets
4	NNs	High accuracy, captures complex patterns	Requires large datasets, computationally expensive	Complex datasets, high-accuracy requirements

The above table shows a comparison highlighting the strengths and weaknesses of each algorithm used in detecting and preventing phishing attacks.

5.2 Key Findings

1. NNs achieved the highest accuracy (94%) and F1-score (92.5%), making them the most effective algorithm for phishing detection in this study. Their ability to model complex, non-linear relationships contributed significantly to their superior performance.
2. RFs also performed exceptionally well, with an accuracy of 92% and an F1-score of 90.5%. Their robustness and ability to hold high-dimensional data make them be strong alternative to NNs. They offered a strong balance between interpretability, computational efficiency, and detection accuracy.
3. SVM achieved competitive results (88% accuracy) but required extensive tuning and computational resources. Optimal performance was only reached after careful selection of the kernel and parameters, which may limit its practicality in real-time systems.
4. DTs were the least effective, with an accuracy of 85%. While interpretable and fast to train, they were prone to overfitting and achieved lower performance compared to ensemble and deep learning models. Despite this, their transparency in decision-making makes them useful in scenarios where explainability is a priority.
5. Visualization of Results highlighted the clear performance gap between deep learning/ensemble methods and simpler models, reinforcing the advantage of more sophisticated approaches in phishing detection.

6. Discussion

The experimental results highlight the significant potential of ML algorithms in detecting and preventing phishing attacks, with NNs and RFs emerging as the top-performing models. Below is a detailed discussion of the results, their implications for cybersecurity, an analysis of the strengths and limitations of each algorithm, and actionable suggestions for improving phishing detection and prevention. The results reveal

that NNs achieved the highest accuracy (94%) and F1-score (92.5%), making them the most effective algorithm for phishing detection in this study. Their capability to capture complex, non-linear patterns of data allows them to identify subtle indicators of phishing that simpler models might miss. This adaptability makes them resilient against new and evolving phishing techniques that deviate from previously known patterns. This makes NNs particularly suitable for integration into real-time cybersecurity systems, such as email filters, web browsers, and network security tools, where high accuracy and minimal false positives are critical.

RFs also performed exceptionally well, producing an accuracy of 92% and an F1-score of 90.5%. As an ensemble method, RFs merge the predictions of multiple DTs, reducing overfitting and improving generalization. They also offer valuable feature importance rankings, which can inform security analysts about the most relevant phishing indicators for targeted defense strategies. Their robustness and interpretability make them a strong alternative to NNs, especially in environments where explainability is important, such as compliance-driven industries or when justifying decisions to stakeholders.

SVM achieved competitive results, with an accuracy of 88%. However, the performance was extremely dependent on the choice of kernel and parameter tuning, and they required significant computational resources. In scenarios where phishing indicators are linearly inseparable, SVM can still provide strong performance if computational constraints are manageable. While effective, SVM may not be the best choice for large-scale or real-time applications due to these constraints.

DTs, while interpretable and fast to train, were the least effective, with an accuracy of 85%. Their susceptibility to overfitting and lower performance compared to ensemble and deep learning models limit their applicability in high-stakes cybersecurity scenarios. However, their transparent decision rules make them useful as an initial detection layer or for educational demonstrations of phishing detection principles. However, they can serve as a baseline model for comparison or in situations where simplicity and interpretability are prioritized over accuracy.

6.1 The Implications of these algorithms for Cybersecurity

- NNs and RFs are highly effective in detecting phishing attacks and can be integrated into real-world cybersecurity systems. Their deployment in multi-layered defense strategies can significantly reduce phishing-related breaches.
- SVM can be used in scenarios where computational resources are not a constraint and high precision is required. This makes them suitable for targeted analysis of high-risk data streams.
- DTs are suitable for baseline models or scenarios where interpretability is more important than accuracy. They can also serve as quick-deployment models in resource-limited environments.
- The adoption of ensemble methods such as RFs and deep learning models (e.g., NNs) can significantly enhance phishing detection capabilities in real-world applications. These methods also adapt better to changes in phishing strategies when retrained regularly.
- These models can be integrated into email security systems, web browsers, and network monitoring tools to provide real-time protection against phishing threats. Integration into cloud-based platforms can further extend protection to distributed and mobile workforces.
- The results underscore the importance of continuous model training and updating to adapt to evolving phishing algorithms, as attackers constantly refine their methods to bypass detection systems. Automated retraining pipelines can ensure that detection systems remain current without manual intervention.

6.2 The Analysis of the Strengths and Limitations of Each ML Algorithm

Algorithm	Strengths	Limitations
DTs	- Interpretable and easy to visualize - Fast to train and suitable for small datasets	- Prone to overfitting - Lower accuracy compared to other models
RFs	- Robust and holds high-dimensional data - It reduces overfitting through ensemble learning	- Computationally intensive for large datasets - Less interpretable than single DTs
SVM	- Effective in high-dimensional spaces - Works well with clear margin separation	- Requires careful tuning of kernel and parameters - Computationally expensive for huge datasets
NNs	- High accuracy and ability to capture complex patterns - Suitable for real-time applications	- Requires large datasets and computational resources - Less interpretable ("black-box" nature

6.3 Suggestions for Improving the Detection and Prevention of Phishing Attacks with ML

To improve phishing detection using ML, suggestions include adopting hybrid models for enhanced accuracy, incorporating advanced features engineering (e.g behavior and temporal features), deploying real-time detection systems (e.g browser extensions and API-based solutions), implementing continuous learning to adapt to evolving threats, leveraging explainable AI (XAI) for transparency, fostering collaboration and data sharing among organizations, integrating user education programs and using adversarial training to improve model robustness against sophisticated attacks. Additionally, building domain-specific datasets that reflect current phishing trends can further enhance detection performance. Regular benchmarking of models against emerging attack techniques will ensure they remain effective over time.

7. Conclusion

The experimental results demonstrate that ML, particularly NNs and RFs, can significantly enhance phishing detection and prevention. These findings align with global cybersecurity research trends, reinforcing the relevance of advanced ML methods in addressing modern threats. These models excel in capturing complex patterns and generalizing to new threats, making them suitable for real-world cybersecurity applications. However, the choice of algorithm should consider factors like computational resources, interpretability, and the specific requirements of the application. For instance, high-traffic enterprise environments may prioritize computational efficiency, while regulated industries may prioritize explainability. By adopting hybrid models, improving feature engineering, and integrating real-time detection systems, organizations can build robust defenses against phishing attacks. Furthermore, continuous learning, explainability, and user education are critical for staying ahead of evolving threats. This research provides a comprehensive roadmap for leveraging ML to combat phishing, offering valuable insights and actionable recommendations for researchers and practitioners in cybersecurity.

Compliance With Ethical Standards

Conflicts of interest: Authors declared that they have no conflict of interest.

Human participants: The conducted research follows ethical standards and the authors ensured that they have not conducted any studies with human participants or animals.

References

[1] I. Goodfellow, Y. Bengio, and A. Courville. “Deep Learning”. MIT Press. 2016.
[2] IBM Security. Machine Learning in Cybersecurity. Retrieved from https://www.ibm.com/security. 2021.
[3] J. Saxe, ans K. Berlin. “Deep Neural Network-Based Malware Detection Using Two-Dimensional Binary Program Features”, IEEE. 2015.
[4] Symantec. “The Role of Machine Learning in Cybersecurity”, Retrieved from https://www.symantec.com. 2019.
[5] N. Abdelhamid, A. Ayesh, and F. Thabtah. “Phishing Detection Based Associative Classification Data Mining,” Expert Systems with Applications, Vol. 41, No. 13, pp. 5948-5959, 2014.
[6] Z. Alkhalil, C. Hewage, L. Nawaf, and I. Khan. “Phishing Attacks: A Recent Comprehensive Study and a New Anatomy”, Frontiers in Computer Science, Vol. 3, pp. 563060, 2021.
[7] L. Breiman, “Random Forests. Machine Learning”, Vol. 45, No. 1, pp. 5-32, 2001.
[8] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. “Classification and Regression Trees”, CRC Press. 1984.
[9] C. Cortes, & V. Vapnik. “Support-Vector Networks”, Machine Learning, Vol. 20, No. 3, pp. 273-297.
[10] J. H. Friedman. “Greedy Function Approximation: A Gradient Boosting Machine,” Annals of Statistics, Vol. 29, No. 5, pp.1189-1232, 2001.
[11] T. Hastie, R. Tibshirani, and J. Friedman. “The Elements of Statistical Learning: Data Mining, Inference, and Prediction”, Springer. 2009.
[12] R. M. Mohammad, F. Thabtah, and L. McCluskey. “Phishing Websites Features”, arXiv preprint arXiv:1901.02385, 2019.
[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. “Learning Representations by Back-Propagating Errors,” Nature, Vol. 323, No.6088, pp. 533-536.
[14] Sahoo, D., C. Liu, and S. C. H. Hoi. “Malicious URL Detection Using Machine Learning: A Survey”, ACM Computing Surveys, Vol. 53, No. 1, pp.1-36, 2020.
[15] Y. Zhang, J. I. Hong and L. F. Cranor. “CANTINA: A Content-Based Approach to Detecting Phishing Web Sites,” In Proceedings of the 16th International Conference on World Wide Web, pp. 639-648, 2018.