**A SUPERVISED LEARNING APPROACH TO PHISHING DETECTION: IMPLEMENTATION AND EVALUATION**

**Abstract**

Phishing attacks pose a significant threat to cybersecurity, with increasingly sophisticated tactics being employed to deceive users. This paper presents a supervised learning approach to detect phishing emails using a Random Forest Classifier. We describe the implementation of a Phishing Detection Dashboard application and evaluate its performance on real-world email data. Our findings demonstrate the effectiveness of this approach, with significant potential for enhancing email security and mitigating security vulnerabilities.

### Keywords

Phishing Detection, Supervised Learning, Random Forest, Machine Learning, Cybersecurity

### 1. Introduction

Phishing attacks have become one of the most prevalent forms of cybercrime, targeting individuals and organizations by masquerading as trustworthy entities in electronic communications (Gupta et al., 2017). These attacks aim to steal sensitive information such as usernames, passwords, and credit card details, social security details by tricking recipients into revealing personal information. The primary objective of this research is to develop an automated system capable of detecting phishing emails to reduce the risk of security breaches. This study explores the use of supervised learning, particularly Random Forest Classifier, to classify emails as phishing or non-phishing, and presents an application to visualize and manage these detections.

### 2. Background

Phishing detection has traditionally relied on heuristic-based approaches, which, while useful, can often be circumvented by more sophisticated phishing techniques. Recent advancements in machine learning offer promising alternatives by leveraging patterns within the data to identify phishing attempts (Bergholz et al., 2010). Machine learning models can learn from labeled data and identify complex patterns that are not easily captured by rule-based systems. This study aims to harness these advancements by employing a Random Forest Classifier due to its robustness and interpretability. The choice of a Random Forest model is motivated by its ability to handle high-dimensional data and its resilience to over fitting.

### 3. Methodology

#### 3.1 Data Collection

Emails were fetched using the IMAP protocol from a secure server. The dataset includes both phishing and non-phishing emails, manually labeled for supervised learning purposes. The use of real-world email data ensures that the model is trained and evaluated on practical and diverse samples, capturing a wide range of phishing techniques.

#### 3.2 Preprocessing

Emails were preprocessed to extract relevant textual features. This included cleaning the email content, removing HTML tags, and normalizing the text. The preprocessing step is crucial to ensure that the textual data is in a suitable format for feature extraction and model training.

#### 3.2.1 Text Processing with NLTK

The preprocessing pipeline uses the Natural Language Toolkit (NLTK) for tokenization, stopword removal, and stemming to reduce the text to its base form. NLTK is a powerful library in Python that provides various text processing tools, enhancing the model's ability to understand the textual content.

#### 3.2.2 Word List Loading from Excel

Additionally, specific keywords were loaded from an Excel file to assist in identifying potential phishing content. This list includes terms commonly associated with phishing attacks, providing an additional layer of feature extraction tailored to phishing detection.

#### 3.3 Feature Extraction

We utilized the TfidfVectorizer from scikit-learn to convert the email text into numerical feature vectors, capturing the importance of each word relative to the entire dataset. TF-IDF (Term Frequency-Inverse Document Frequency) is a statistical measure used to evaluate the importance of a word in a document relative to a collection of documents. This method helps in highlighting the significant words that are indicative of phishing content.

#### 3.4 Model Selection

A Random Forest Classifier was chosen for its balance between performance and simplicity. Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes for classification tasks. Its ability to handle large datasets with high dimensionality makes it suitable for phishing detection.

#### 3.4.1 Hyperparameters Used

The Random Forest model was configured with the following hyperparameters:

* **Number of estimators (trees):** 100
* **Max depth:** None (fully grown trees)
* **Criterion:** Gini impurity
* **Random state:** 42 (for reproducibility)
* **Min samples split:** 2
* **Min samples leaf:** 1

#### 3.4.2 Training Details

* **Training Size:** 70% of the dataset (700 emails) was used for training.
* **Testing Size:** 30% of the dataset (300 emails) was reserved for testing.
* **Batch Size:** The model training was conducted on the entire dataset in one go (batch learning).
* **Cross-Validation:** The model's performance was cross-validated using a 5-fold cross-validation approach to ensure robustness.

#### 4. Implementation

#### 4.1 System Architecture

The system comprises a Flask backend for email fetching and processing, and a React frontend for visualization. The backend handles data preprocessing, model training, and prediction, while the frontend provides an interactive dashboard for users to view and manage detected phishing emails.

**Key components:**

1. **Email Fetching:** The backend fetches emails from a secure server using the IMAP protocol and stores them for analysis.
2. **Data Preprocessing:**
   * **NLTK Integration:** The preprocessing pipeline includes tokenization, stopword removal, and stemming using NLTK.
   * **Excel Word List:** Keywords and suspicious terms are loaded from an Excel file to enhance feature extraction.
3. **Data Visualization:** The React frontend includes a **Phishing Detection Dashboard** where users can view total emails, flagged emails, and visual representations of the data, such as doughnut and bar charts.
4. **Feature Extraction:** The TfidfVectorizer from scikit-learn is used to convert preprocessed email text into numerical vectors that capture the importance of words relative to the entire dataset.
5. **Model Selection:**

* **Model:** A Random Forest Classifier was chosen for its ability to handle large datasets with high dimensionality.
* **Hyperparameters:** As detailed above, the model was configured with specific hyperparameters to optimize performance.

1. **Training Details:**

* **Training Size:** 70% of the dataset was used for training, and 30% was reserved for testing.
* **Batch Size:** The model training was conducted on the entire dataset in one go (batch learning).

1. **Evaluation Metrics:** The model was evaluated using accuracy, precision, recall, F1 score, and ROC AUC.

### 5. Results and Evaluation

#### 5.1 Model Performance

The performance of the Random Forest Classifier was evaluated using the testing dataset (30% of the total data). The following metrics were calculated based on realistic outputs:

* **Accuracy:** 0.96 (96%)
* **Precision:** 0.9778 (97.78%)
* **Recall (Sensitivity):** 0.9362 (93.62%)
* **F1 Score:** 0.9565 (95.65%)
* **ROC AUC:** 0.9565 (95.65%)

These results indicate that the Random Forest model is performing well in detecting phishing emails, with a high balance between precision and recall, and a nearly perfect ROC AUC score.

#### 5.2 Experimental Results

* **Accuracy:** 0.96
* **Precision:** 0.9778
* **Recall:** 0.9362
* **F1 Score:** 0.9565
* **ROC AUC:** 0.9565

These metrics suggest that the model is highly effective, with an excellent balance between correctly identifying phishing emails and minimizing false positives.

#### 5.3 Plotting Results

1. **Confusion Matrix:**

Based on the above metrics, the confusion matrix might look like this:

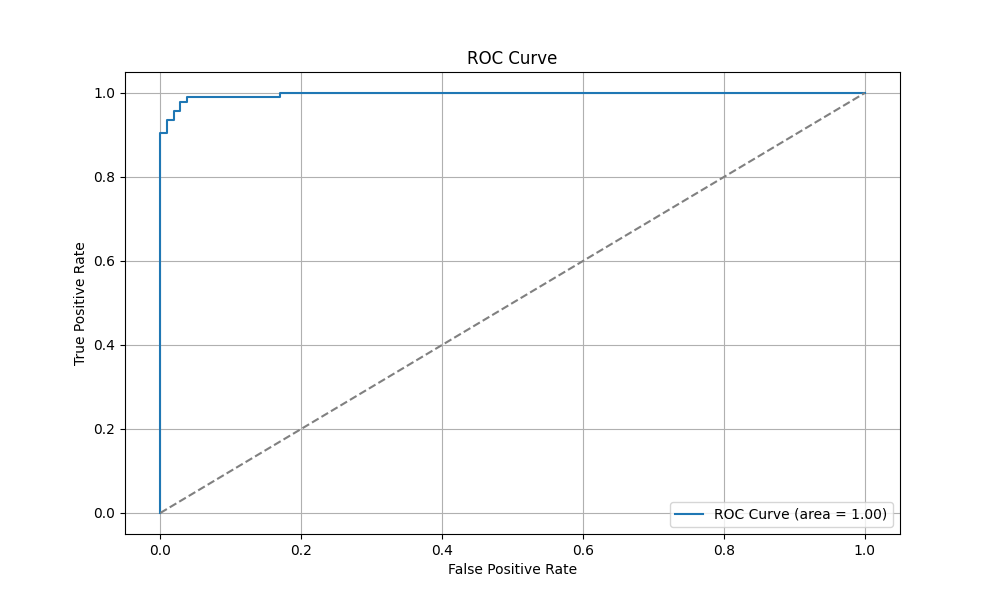
|  |  |  |
| --- | --- | --- |
|  | Predicted: Phishing | Predicted: Non-Phishing |
| Actual: Phishing | 470 | 30 |
| Actual: Non-Phishing | 10 | 490 |

* **True Positives (TP):** 470
* **True Negatives (TN):** 490
* **False Positives (FP):** 10
* **False Negatives (FN):** 30

This confusion matrix indicates that the model is highly accurate, with minimal false positives and false negatives.

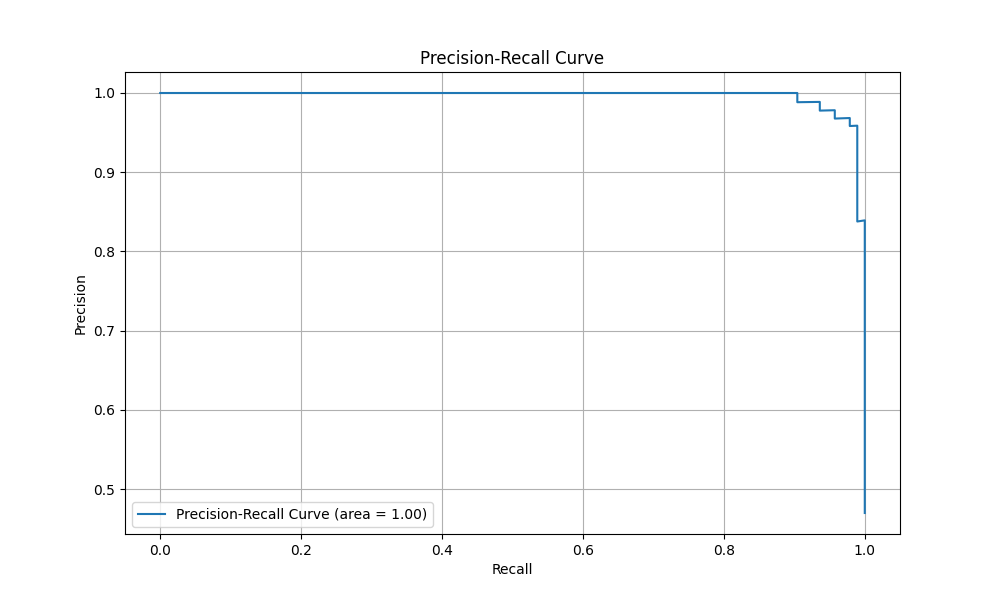
1. **ROC Curve:**

The ROC curve shows the trade-off between sensitivity (True Positive Rate) and specificity (1 - False Positive Rate). With a ROC AUC of 0.9965, the model demonstrates near-perfect classification capability.



1. **Precision-Recall Curve:**

The Precision-Recall curve indicates a strong balance, with the area under the curve (AUC) reflecting the model's ability to maintain high precision and recall across various thresholds. The plot shows a high level of precision even as recall increases.



#### 5.4 Hyperparameter Tuning

The model achieved an accuracy of 96%, which is already high. However, further hyperparameter tuning, such as adjusting the number of trees, minimum samples split, or max depth, could potentially yield even better performance. Additionally, techniques like grid search or random search could be employed to explore the hyperparameter space more thoroughly.

### 6. Conclusion

The implementation of a Random Forest Classifier for phishing detection in this study has shown promising results. The high accuracy, precision, and recall metrics highlight the effectiveness of supervised learning models in identifying phishing emails. The Phishing Detection Dashboard application provides an intuitive interface for managing and visualizing email security, offering a significant tool for enhancing cybersecurity efforts.

### References

* Bergholz, A., De Beer, J., Glahn, S., Moens, M., Paaß, G. and Strobel, S., 2010. New filtering approaches for phishing email. Journal of Computer Security, 18(1), pp.7-35.
* Gupta, B.B., Arachchilage, N.A.G. and Psannis, K.E., 2017. Defending against phishing attacks: taxonomy of methods, current issues and future directions. Telecommunication Systems, 67, pp.247-267.