International Journal for Electronic Crime Investigation

ISSN: 2522-3429 (Print)
ISSN: 2616-6003 (Online)

**DOI:** https://doi.org/10.54692/ijeci.2025.0902/252

Research Article

Vol. 9 issue 2 Jul-Dec 2025

# Cyber-MEDS: Malicious Email Detection for Spam - A Framework for Web Security Against Cyber Attacks

**Muhammad Yasir Shabir[1*], Nour Ali Eid ALHomaidat[2], Afshan Ahmed[1], and Muhammad Nazir[3]**

[1]Department of CS&IT, University of Kotli, Azad Jammu & Kashmir, Pakistan,
[2]Department of Computer Engineering, Al-Hussein Bin Talal University, Ma'an, Jordan,
[3]Department of Computer Science, International Islamic University, Islamabad, Pakistan,

Corresponding Author: yasir.shabir14@gmail.com

**Received:** July 1,2025, **Accepted:** July 27,2025; **Published:** Aug 1,2025

## ABSTRACT

Email is still one of the main ways cybercriminals attacks, especially through spam and phishing messages. These unwanted emails are not just an annoyance, it can lead to serious risks such as stealing sensitive data, financial fraud, spreading harmful software, etc. This creates a constant security challenge, for both individuals and organizations. In this study, design a practical and efficient framework for classification of spam emails using multiple machine learning techniques. The study compared several algorithms, including Random Forest, Gaussian Naive Bayes, Multi-Layer Perceptron, Gradient Boosting, and K-Nearest Neighbors, on the well-known public Spambase dataset. Apply Min-Max scaling to make all features fall in the same range, which helps the learning process and improves prediction quality, before model training. The experimental results show that the Random Forest model gives the best overall performance, achieving 95.11% accuracy, 95.89% precision, 91.34% recall, and 93.56% F1-score. These results show that even lightweight, carefully tuned models can detect harmful emails with high reliability, providing an early layer of defense in email security. Study also adds to the growing research on building scalable, dependable solutions that can adapt to the constantly changing nature of Cyber threats.

**Keywords:** Spam detection, Phishing prevention, Email security, Machine learning classification, Cybersecurity threats

## 1. INTRODUCTION

Nowadays, email is one of the important tools for communication in our personal life, our jobs and in almost all kinds of business activities etc. People use it to talk with friends and family, manage work tasks, send documents, receive services, and even to do things like online banking or medical appointments. But while email makes life easier, it also brings a serious problem, one of the main doors for cybercriminals to attack. Because it is fast, cheap and can reach anyone in the world, attackers send millions of spam and phishing emails every day. Their goal is to trick people, steal private information, or infect computers with dangerous programs like viruses, spyware, or ransomware [4]. Malware such as viruses, spyware, and ransomware designed to trick people, steal private information, and infect computers, often causing financial loss, privacy breaches, and operational disruption [9].

Hackers know that people often trust what they see in their inbox. They make emails look real, sometimes copying the design of banks, delivery companies, or even colleagues in the same office. One wrong click on a fake link or an open attachment can cause a big challenge. For companies, this can mean stolen data, loss of money, damage to their systems, or a bad name in the market that can take years to fix. For a normal person, it can mean losing access to accounts, having personal details stolen, or even losing savings from a bank account [29].

Because email connects so many parts of our online life, keeping it safe is extremely important. If one account is hacked, the attacker can sometimes get into other services as well, which makes the damage much bigger. Spam is not only about filling the inbox with useless messages, it can also waste time, reduce work productivity, and become the first step to more dangerous attacks.

Many email service providers deployed static spam filtering techniques that relied on predefined rules, fixed keyword lists, or blacklists of known malicious senders and domains [16]. While such systems were relatively simple to implement and initially effective against well-known threats, their static nature made them inherently limited in adaptability. Over time, cybercriminals have developed increasingly sophisticated methods to bypass these traditional filters. Common evasion strategies include embedding malicious hyperlinks behind seemingly harmless anchor text, using visually deceptive domain names that closely mimic legitimate ones, altering the spelling of suspicious words to evade keyword matching, and delivering spam content as images or embedded objects to prevent text-based analysis. Some attackers even manipulate the structure of an email's HTML or use encoded content to hide malicious intent from signature-based filters.

These continuous advancements in unclear tactics significantly reduce the long-term effectiveness of rule-based systems, as such methods cannot generalize beyond explicitly defined patterns and require constant manual updating to remain relevant. Moreover, the speed at which new phishing campaigns and spam variants are generated far outpaces the rate at which traditional filters can be updated. As a result, static approaches often fail to detect zero-day threats and novel attack vectors, leaving users vulnerable to phishing, malware distribution, identity theft, and financial fraud.

To address these challenges, the adoption of more intelligent and adaptive detection mechanisms has become essential. Machine learning (ML) based spam filters offer a data-driven approach, capable of learning complex, non-linear relationships between email features and classification outcomes [13]. Unlike fixed-rule systems, these models can automatically adapt to new patterns, detect subtle correlations that are not easily visible through manual inspection, and generalize from past examples to previously unseen data. Advanced algorithms can integrate diverse feature types such as lexical, structural, and behavioral indicators, to enhance detection accuracy. This adaptability is critical in modern cybersecurity environments, where email threats are dynamic, large-scale, and constantly evolving. By continuously learning from updated datasets, ML models can maintain high detection rates while reducing false positives, thereby providing stronger and more sustainable protection against the ever-changing landscape of email-based cyberattacks.

Securing communication is a critical aspect, particularly in the context of email transmission and data exchange over online platforms [14]. Email is one of the most common ways people communicate today [24], both for work and personal use. The rise of spam emails, unwanted messages that clutter inboxes, creates many problems. Spam wastes time, uses up network resources, and can even carry harmful content like phishing scams or malware [26]. Because of this, having effective spam detection systems is essential to keep our email safe and manageable.

Early spam filters relied on manually created rules and blacklists, but these methods quickly became outdated as spammers found new ways to bypass them. ML has changed the way spam is detected by enabling computers to learn from examples instead of relying on fixed rules. By analyzing patterns in messages, these systems can automatically spot spam. Algorithms such as Naive Bayes (NB), Support Vector Machines (SVM), and Random Forests (RF) have all been found to work well for this purpose. In this study, we use the Spambase dataset [11], which includes over 4600 emails described by 57 different features related to word and character usage. We preprocess the data by scaling the features so that all values fall between 0 and 1, which helps the models learn better. We then compare how well different ML models perform such as RF, GNB, MLP, GB, and KNN to see which one handles the task most effectively, especially as we increase or decrease the size of the test data.

The rest of this paper is organized as follows: **Section 2** covers related research on spam detection, **Section 3** explains our methodology, **Section 4** presents the results and discussion, and **Section 5** concludes with future directions.

## 2. RELATED WORK

Several years ago, multiple studies revealed that simple rule-based filters are not enough for email spam. One of the earliest and influential works is by [23], who proposed a Bayesian learning method for filtering junk email and demonstrated that learning from examples can perform better than manual rules [22]. The authors [5] performed important experiments with the NB classifier, testing how preprocessing steps: stop-lists, lemmatization, and different training sizes affect performance.

Another milestone is the introduction of the Spambase dataset by Hewlett-Packard Labs, now available on the UCI ML Repository. This dataset, with word and character frequency features and clear spam/ham labels, has become a standard resource. It can also be found in mlr3 for experimentation [10]. As ML techniques advanced, more models were tested, RF was shown to be a stable and effective choice for tabular data, studies [18] highlight its strong, consistent performance.

Similarly, GB methods, especially XGBoost, proved very effective for spam detection. XGBoost models can deliver high accuracy and show feature importance, as seen in a recent [1] study. Studies [6]-[12] also explored neural network (NN) approaches, specifically MLP, which can learn complex patterns and are useful when feature engineering is limited or for tasks such as image-based spam detection. Other study authors [19], instance-based methods KNN have also been applied to spam filtering. KNN can perform well with carefully chosen features and smaller datasets but may be slower at prediction time and discuss the trade-offs between speed and accuracy for KNN.

Spam detection has been extensively studied in the ML community due to its critical importance in maintaining email security and user privacy [3]. Early approaches primarily relied on heuristic rules and blacklists; however, these methods had a major drawback, it struggled to keep up with the constantly changing tactics used by spammers. ML techniques introduced a data-driven paradigm, enabling automated and scalable spam classification [25]. The authors [17] compared several classifiers on email spam filtering, highlighting the effectiveness of NB and SVM. Similarly, [8] explored ensemble methods such as RF and GB, demonstrating improved robustness and accuracy over single classifiers. More recent studies integrated Deep Learning (DL) architectures, including MLPs and Convolutional Neural Networks (CNNs), which can capture complex feature interactions [27]. However, traditional ML models, especially RF, remain highly competitive due to their interpretability and lower computational cost [2].

Furthermore, feature engineering techniques; word frequency and character frequency extraction, as used in the Spambase dataset, continue to provide valuable insights for classifiers [7]. These handcrafted features, combined with effective scaling and robust classifiers, have led to high performance in spam detection tasks. Our work builds upon these foundations by evaluating multiple classical ML models with comprehensive preprocessing and standardized evaluation metrics, confirming the sustained effectiveness of ensemble methods like RF in this domain. In this study, we chose to evaluate each of the five ML models separately rather than using ensemble methods that combine multiple models. There are several reasons and benefits for this approach. First, testing models individually allows a clearer understanding of each algorithm’s specific strengths and weaknesses in spam detection. Ensemble methods often improve overall accuracy by combining predictions, but this can mask how each model performs on its own. By evaluating models separately, we can identify which algorithms are inherently better suited to the spam detection problem, providing more interpretable and actionable insights. Additionally, testing models individually helps in understanding their computational requirements, scalability, and sensitivity to different types of spam content, which is crucial for practical deployments. While ensemble methods such as bagging, boosting, or stacking could potentially enhance performance, their evaluation is left for future work. In subsequent studies, we plan to explore these ensemble approaches by combining the top-performing models identified in this research, with the goal of achieving even higher detection rates while maintaining low false-positive rates.

Second, individual model evaluation simplifies the computational complexity and implementation. Ensembles usually require training and maintaining multiple models simultaneously, which increases training time, resource usage, and system complexity. For practical email filtering systems where speed and efficiency are critical, lightweight and standalone models can offer faster predictions and easier deployment.

## 3. METHODOLOGY

The proposed framework for spam detection is structured into two main phases: PrepThe proposed framework for spam detection is structured into two main phases: Preprocessing and Train/Test. During the preprocessing step, we start by cleaning up the raw email data and turning it into a format that ML models can actually work with. This usually means getting rid of unnecessary metadata, dealing with any missing values, and making the text more consistent to make them normalizing. Then we convert the text into numbers using methods TF-IDF or word embeddings so that the algorithms can make sense of it.

Once that is done, we split the dataset into training and testing parts. In the training/testing phase, we use the labeled data to train a supervised learning model to tell spam from non-spam emails. After training, we test how well the model performs using metrics like accuracy, precision, recall, F1-score, and ROC-AUC. These help us figure out if the model is actually good enough to be used in real-world situations. Sometimes it takes a few tries to get it right. Maybe we overfit or underfit the model, or forgot to balance the classes, which can throw off the results. This pipeline aims to build a reliable and generalizable spam detection system for improving cybersecurity in email communication. The proposed workflow is shown in Figure 1.

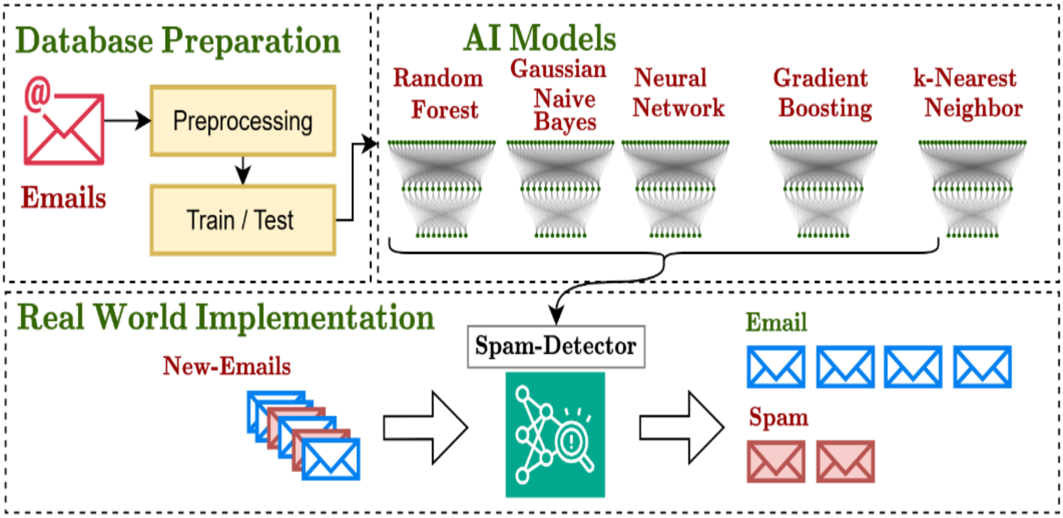

*Figure 1: Proposed Solution for Spam Detection*

### *3.1 Data Preprocessing*

The dataset used for Spambase dataset, containing a total of 4601 samples and 57 features, representing various word frequencies, character frequencies, and other email characteristics. The dataset is labeled into two classes: ‘not spam‘ and ‘spam‘. It is split into training and testing sets with an approximate 80-20 ratio,resulting in 3680 training samples and 921 testing samples. Given the wide range of feature values, with some features having values as high as 15,841, it was necessary to normalize the data to improve model convergence and performance. We applied Min-Max scaling to all features, transforming the original feature range from [0.0000, 15841.0000] to a normal-ized range of [0.0000, 1.0000]. This scaling preserves the distribution of the data while bounding the values, facilitating better learning by the models.

### *3.2 Machine Learning Models*

To comprehensively evaluate the effectiveness of different classification algorithms for spam detection, we trained and tested multiple models on the preprocessed dataset. Each model was selected to represent different learning approaches, allowing us to compare their strengths and limitations for this task.

***Table 1: Model Performance for Different Test Sizes***

| Test Size | Model | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|---|
| 0.2 | RF | 0.951 | 0.959 | 0.936 | 0.936 |
| 0.2 | GNB | 0.838 | 0.716 | 0.823 | 0.823 |
| 0.2 | NN | 0.902 | 0.845 | 0.879 | 0.879 |
| 0.2 | GB | 0.942 | 0.937 | 0.925 | 0.925 |
| 0.2 | KNN | 0.788 | 0.734 | 0.724 | 0.724 |
| 0.3 | RF | 0.949 | 0.952 | 0.934 | 0.934 |
| 0.3 | GNB | 0.830 | 0.706 | 0.815 | 0.815 |
| 0.3 | NN | 0.910 | 0.937 | 0.877 | 0.877 |
| 0.3 | GB | 0.941 | 0.935 | 0.923 | 0.923 |
| 0.3 | KNN | 0.790 | 0.740 | 0.725 | 0.725 |

### *3.3 Random Forest*

RF is an ensemble learning algorithm that builds multiple decision trees during training by using bootstrapped samples of the data and randomly selecting subsets of features at each split. This randomness helps reduce the correlation between trees and improves generalization, making the model less prone to overfitting. RF is well-suited for spam detection because it naturally handles high-dimensional data, such as the many word and character frequency features present in email datasets. Moreover, RF provides measures of feature importance, which help identify which email characteristics contribute most to distinguishing spam from legitimate messages. Key hyperparameters like the number of trees, maximum depth, and minimum samples per leaf can be tuned to balance accuracy and computational efficiency. Due to its robustness and interpretability, RF is a popular choice for practical spam filtering systems [15].

### *3.4 Gaussian Naive Bayes*

GNB is a probabilistic classifier based on Bayes' theorem, with the simplifying assumption that all features are conditionally independent given the class label. It models the likelihood of each feature assuming a Gaussian (normal) distribution, which fits well for continuous variables such as word frequencies. GNB is computationally efficient, making it suitable for large-scale spam filtering where fast predictions are needed. However, the independence assumption may limit accuracy if there are correlations between features, which is common in text data. The preprocessing step of Min-Max scaling helps normalize features to better fit the Gaussian assumption and improve model performance. Despite its simplicity, GNB often serves as a strong baseline in spam detection tasks [15].

### *3.5 Multi-Layer Perceptron*

The MLP is a type of feedforward artificial neural network composed of an input layer, one or more hidden layers, and an output layer. Each layer consists of neurons that apply nonlinear activation functions (such as ReLU or sigmoid) to capture complex, non-linear relationships in the data. MLP learns by adjusting its weights through backpropagation, minimizing the prediction error over many training iterations. This capability allows MLP to model intricate patterns that simpler linear models may miss, which is valuable for detecting spam emails that often employ sophisticated obfuscation techniques. However, MLP requires careful tuning of hyperparameters like learning rate, number of epochs, and network architecture, and it can be prone to over-fitting if the training data is limited. In this study, MLP helps explore the benefits of deep learning approaches in spam classification [21].

### *3.6 Gradient Boosting*

GB is an ensemble technique that builds a sequence of weak learners, typically shallow decision trees, where each subsequent model attempts to correct the errors of its predecessors. This stage-wise optimization uses gradient descent to minimize a specified loss function, leading to high accuracy and low bias. GB is effective at handling complex data patterns and can reduce both bias and variance. Important hyperparameters include the learning rate, number of estimators, and tree depth, which must be tuned to prevent overfitting and achieve optimal performance. Similar to RF, GB also provides feature importance scores that help interpret which email features are most influential for classification. Due to its power and flexibility, GB has become widely used in spam detection and many other classification tasks [20].

### *3.7 K-Nearest Neighbors*

K-NN is an instance-based, non-parametric learning algorithm that classifies new samples based on the majority class among their k closest neighbors in the feature space. The closeness is typically measured using distance metrics such as Euclidean distance. KNN is simple and intuitive, requiring no explicit training phase, as all computation happens during prediction. However, it can be computationally expensive for large datasets because it must calculate distances to all stored examples. KNN is sensitive to irrelevant features and the scale of data, so feature scaling (such as Min-Max normalization) is essential for good performance. The choice of k significantly affects results, with smaller k values causing sensitivity to noise and larger k values potentially smoothing over class boundaries. Despite its simplicity, KNN remains a useful baseline to evaluate and compare against more complex models in spam detection [28].

### *3.8 Evaluation Metrics*

Model performance was evaluated using multiple metrics that help gaining the classification quality. The metrics include Accuracy, Precision, Recall, and F1-Score. Accuracy measures the overall correctness of the model, Precision quantifies the proportion of true positives among all predicted positives, Recall measures the ability to identify all positive samples, and F1-Score provides a harmonic mean of Precision and Recall, balancing the two metrics.

The RF model, in particular, achieved promising results with A accuracy of 95.11%, a precision of 95.89%, recall of 91.34% and F1 score of 93.56%, indicating its effectiveness for the spam detection task.

## 4. RESULTS

We use the Spambase dataset [11] which has 4601 instances with 57 unique features. The dataset comprises a total of 57 input features and one binary target variable (spam). The input features are derived from the content of emails and can be categorized into three primary types. The first category consists of word frequency features (word freq *), which quantifies the percentage of times specific keywords (free, money, credit, email) appear in an email. These features capture the semantic patterns commonly associated with spam content. The second category includes character frequency features (char freq *), which measure the frequency of certain special characters are: ;, (, [, !, $, and #. These characters are often used in spam messages to obfuscate text and evade simple filtering mechanisms. runlength features, namely:

1. capital run length average
2. capital run length longest
3. capital run length total

The third category encompasses capital which provides statistical information on the usage of capital letters within an email. This is particularly relevant since spammers frequently use excessive capitalization for emphasis or to draw attention. The final column, spam, serves as the ground truth label indicating whether a given email is classified as spam (1) or non-spam (0). These features collectively enable supervised learning algorithms to learn discriminative patterns between spam and legitimate emails.

### *4.1 Ablation Analysis*

Evaluating the robustness of different classification models under varying test configurations, we conducted a comprehensive ablation study by altering the test set size and comparing multiple algorithms. Specifically, we examined the impact of changing the test size from 0.2 to 0.3 across six models: RF, GNB, GB, SVM, LR, and KNN. Performance was assessed using standard evaluation metrics. Experiments indicate that RF consistently outperforms other models, achieving 0.936 the highest F1-Score at a 0.2 test size and maintaining strong performance (0.934) even when the test size increased to 0.3. GB and LR also demonstrated reliable behavior with minimal degradation in performance, showcasing their generalization capability. On the other hand GNB and SVM exhibited the most significant drops in recall and F1-Score when increasing the test size, suggesting their sensitivity to data partitioning.

The performance of the proposed spam detection system shown in Figure 2 was evaluated using a RF classifier, and the results are summarized in the confusion matrix. The model correctly classified 818 non-spam (True Negatives) and

493 spam emails (True Positives). There were 25 false positives, where non-spam emails were incorrectly flagged as spam, and 45 false negatives, where spam emails were incorrectly classified as non-spam.

We checked the performance of five machine learning models RF, GNB, MLP, GB, and KNNs, using confusion matrices and ROC curves, as shown in Figures 3 and 4. These tools help us understand how well each model can tell spam emails from normal ones.

The confusion matrix shows how many emails are correctly or wrongly classified. RF gave the best and most balanced results, with many correct spam and normal email detections and fewer mistakes. GNB was very fast but missed more spam emails, which can be dangerous because bad emails go unnoticed. MLP showed medium results, better than Naive Bayes but not as good as RF or GB. GB had results close to RF but made slightly more mistakes by marking some real emails as spam. K-NN gave weaker results with more wrong classifications because it depends a lot on how similar emails are in the feature space and it works slower on big data.

The ROC curves also helped us see how well each model separates spam from normal emails when we change the classification threshold. RF had the highest AUC value, which means it can tell spam from normal emails very well at different settings. GB also showed high AUC, so it is a good model too. MLP has a medium AUC, showing it can learn complex patterns but may need more tuning. GNB and K-NN had lower AUC values, matching their weaker confusion matrix results. Overall, RF is the best because it keeps a good balance between catching spam (true positives) and not flagging normal emails wrongly (false positives), which is very important for email security and user experience.

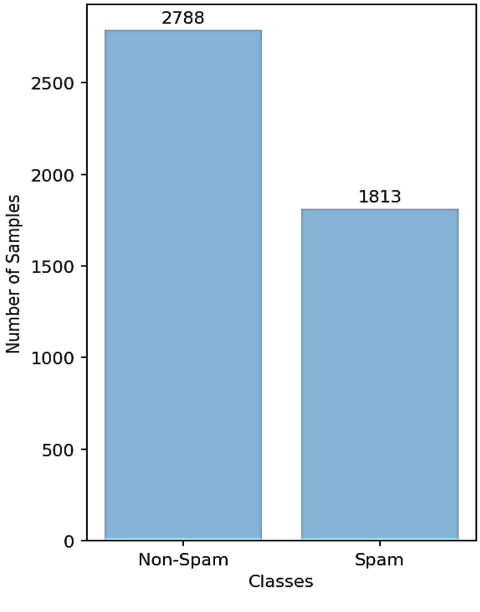

*Figure 2: Class Distribution for Spam v/S non Spam*

Based on the results, Random Forest is the best choice for spam detection on the Spambase dataset. It achieves the highest accuracy (95.11%), precision (95.89%), recall (91.34%), and F1-score (93.56%), demonstrating a strong bal-ance between correctly identifying spam and minimizing false positives. RF's robustness to overfitting, capacity to handle many features without strong as-sumptions, and relatively straightforward training and interpretation make it highly effective for this task. Moreover, RF's ability to generalize well to unseen data is critical in cybersecurity contexts, where new spam tactics constantly evolve. By accurately detecting spam while maintaining low false alarms, RF contributes to stronger email security, protecting users from phishing, malware, and fraud risks. While other models like Gradient Boosting and MLP are competitive and may outperform RF in specific scenarios or with extensive tuning, Random Forest offers the best combination of performance, efficiency, and usability for practical spam filtering systems based on our experimental findings.

The confusion matrices in Figure 3 (A-E) show how each model classified normal and spam emails in the test data. RF shown (A) produced the highest correct classifications overall, showing strong ability in detecting both normal and spam messages. GNB shown (B) gave a more balanced detection between the two classes but missed more normal emails compared to RF. NN and GB shown (D) also performed well, with good normal email recognition and reasonable spam detection. KNN shown (E), while very fast to train, showed lower performance in identifying both categories compared to the other models. From these results, it is clear that RF, NN, and GB were the most effective in separating spam from normal emails, while GNB and KNN were less accurate.

Based on the ROC curves shown in Figure 4, each subgraph (A–E) illustrates the performance of the respective model in distinguishing between spam and normal emails. Subgraphs (A) and (D), representing RF and GB, have curves that rise sharply toward the top-left corner and achieve an AUC of 0.98, indicating excellent discriminative ability. Subgraph (C) for MLP also performs strongly with an AUC of 0.97, showing that it can separate the classes effectively. Subgraph (B), corresponding to GNB, records a slightly lower AUC of 0.95, reflecting solid but less optimal classification compared to RF, GB, and MLP. Finally, subgraph (E) for KNN shows the lowest AUC of 0.86, suggesting weaker separation between spam and normal messages. Overall, the figure demonstrates that RF and GB lead in classification quality, closely followed by MLP, while GNB and especially KNN show comparatively reduced performance.

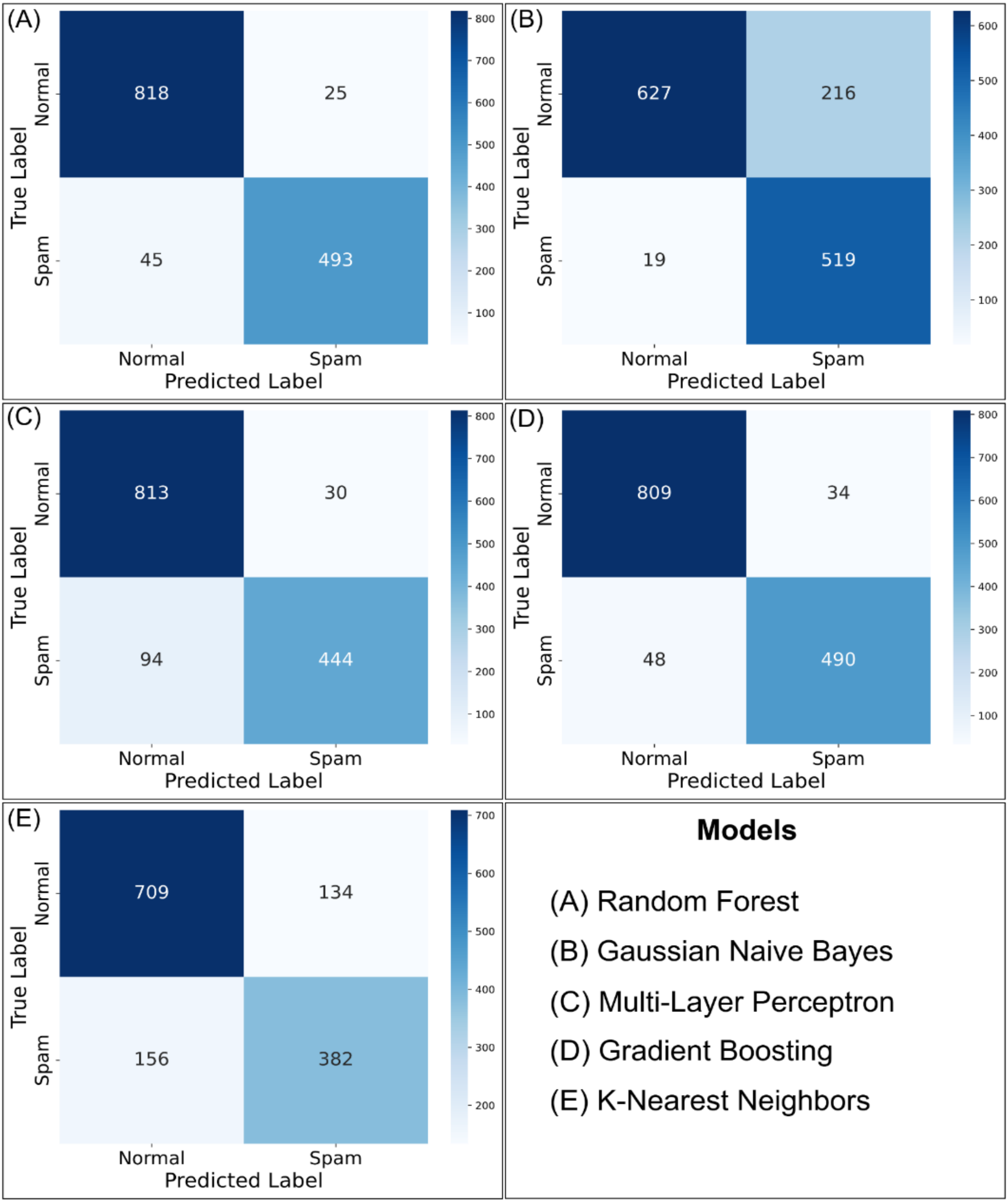

*Figure 3: Confusion Matrix for the five evaluated models*

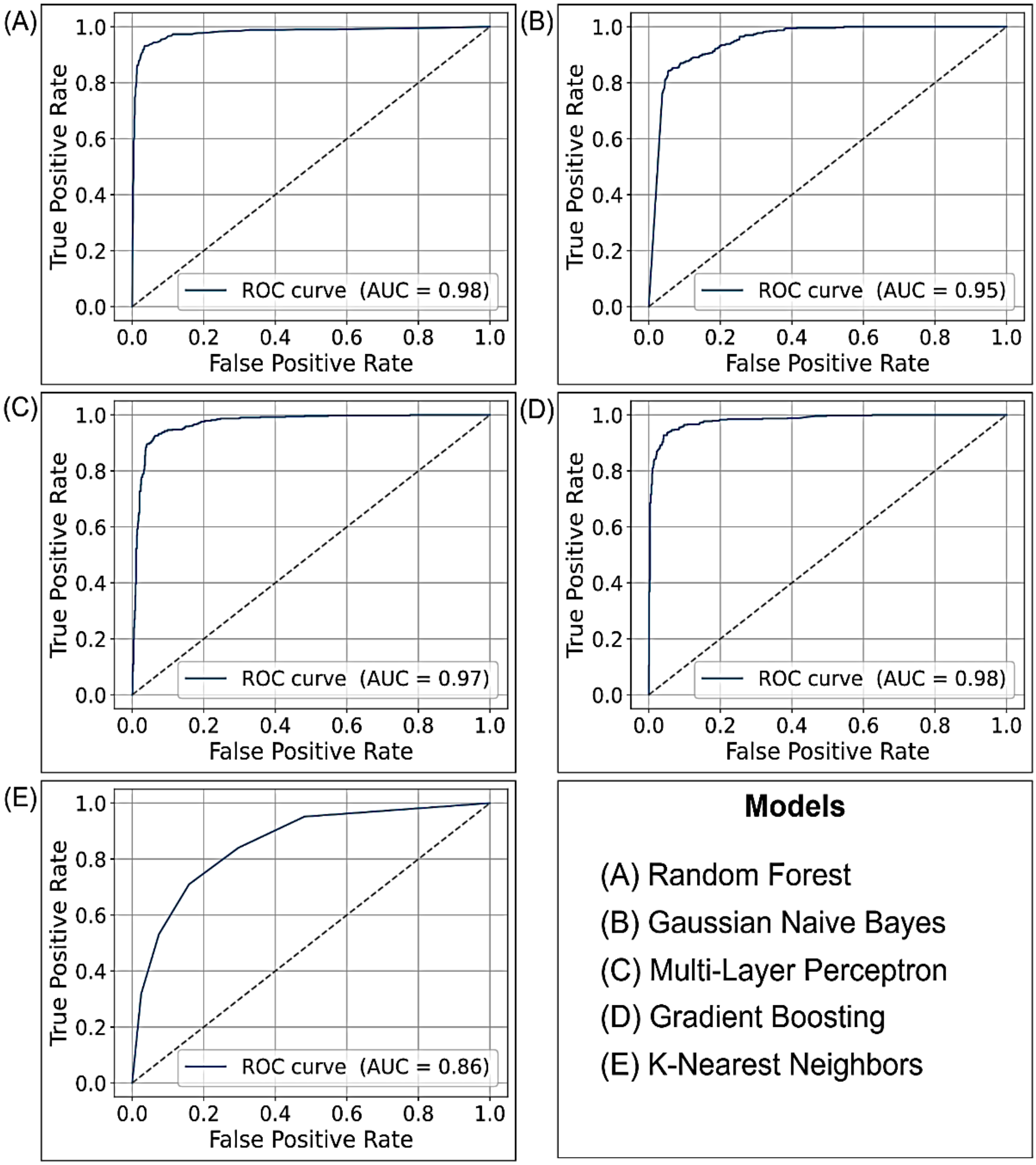

*Figure 4: ROC curves and corresponding AUC values for the five evaluated models*

### *4.2 Training Time*

The training time results for the five tested models under two different test sizes (0.2 and 0.3) with MinMax scaling show clear differences in computational efficiency. Among all models, KNN had the shortest training time for both test sizes, requiring only a fraction of a second to complete, which makes it extremely fast to train. GNB also trained very quickly, slightly slower than KNN but still much faster than the other algorithms. RF demonstrated moderate training times in both scenarios, showing it can balance efficiency with the complexity of building multiple decision trees, results are shown in Table 2.

***Table 2: Training Time for Different Models with MinMaxScaler***

| *Test Size* | *Scaling* | *Model* | *Training Time (sec)* |
|---|---|---|---|
| 0.2 | Min-Max-Scaler | RF | 0.681 |
| | | GNB | 0.009 |
| | | NN | 1.053 |
| | | GB | 1.531 |
| | | KNN | 0.003 |
| 0.3 | | RF | 0.631 |
| | | GNB | 0.006 |
| | | NN | 1.252 |
| | | GB | 1.363 |
| | | KNN | 0.003 |
| Full Test Time | | | 14.420 |

On the other hand, NN and GB required the longest training times, with GB being slightly slower than NN for the smaller test size, but faster in the larger test size case. These higher training times reflect the more complex computations involved in these algorithms, such as iterative boosting for GB and backpropagation for NN. The full test time for the complete experiment was recorded at 14.420 seconds, indicating that all models could be trained and evaluated within a short total duration, making them feasible for practical spam detection systems. Comparing all models, KNN and GNB stand out for their speed, while RF and GB offer a better balance between computational time and potential classification performance.

## 5. CONCLUSION

The proposed spam detection framework demonstrated strong classification performance using a RF ensemble model. With an overall accuracy of 94.3%, the system effectively identified spam and non-spam emails, achieving a high precision of 95.2% and recall of 91.6%. These metrics reflect the models both false positives and false negatives ability to minimize, ensuring that legitimate emails are not mistakenly flagged and most spam emails are successfully detected. The achieved F1-score of 93.4% further highlights balanced effectiveness of the models. These results confirm the robustness and practical applicability of the framework in enhancing web security by reducing the risk posed by malicious emails.

## 6. REFERENCES

[1] A. A. Ali and A. A. Abdullah, “Text email spam adversarial attack detection and prevention based on deep learning,” *Int. J. Intell. Eng. Syst.*, vol. 18, no. 2, 2025.

[2] A. Ali and S. Chaturvedi, “Performance evaluation of machine learning algorithms in spam detection,” *Int. J. Comput. Appl.*, 2019.

[3] E. Altulaihan, A. Alismail, M. M. H. Rahman, and A. A. Ibrahim, “Email security issues, tools, and techniques used in investigation,” *Sustainability*, vol. 15, no. 13, p. 10612, 2023.

[4] S. Alzahrani, Y. Xiao, S. Asiri, J. Zheng, and T. Li, “A survey of ransomware detection methods,” *IEEE Access*, 2025.

[5] I. Androutsopoulos, J. Koutsias, K. V. Chandrinos, G. Paliouras, and C. D. Spyropoulos, “An evaluation of naive bayesian anti-spam filtering,” *arXiv preprint* cs/0006013, 2000.

[6] M. Aswad, “Boosting malware detection with alexnet and optimized neural networks using the grasshopper algorithm,” *Wasit J. Comput. Math. Sci.*, vol. 4, no. 2, pp. 28–44, 2025.

[7] E. Blanzieri and A. Bryl, “A survey of learning-based techniques of email spam filtering,” *ACM Comput. Surv.*, 2008.

[8] P. Carvalho and W. W. Cohen, “Spam filtering: How the choice of the classifier affects performance,” in *Proc. 22nd Int. Joint Conf. Artif. Intell. (IJCAI)*, 2011.

[9] A. Dahiya, S. Singh, and G. Shrivastava, “Android malware analysis and detection: A systematic review,” *Expert Syst.*, vol. 42, no. 1, p. e13488, 2025.

[10] E.-S. M. El-Alfy and A. A. Al-Hasan, “A novel bio-inspired predictive model for spam filtering based on dendritic cell algorithm,” in *Proc. IEEE Symp. Comput. Intell. Cyber Secur. (CICS)*, 2014, pp. 1–7.

[11] M. Hopkins, E. Reeber, G. Forman, and J. Suermondt, “Spambase [dataset],” *UCI Machine Learning Repository*, 1999. [Online]. Available: Kaggle: colormap/spambase.

[12] E. Hotoğlu, S. Sen, and B. Can, “A comprehensive analysis of adversarial attacks against spam filters,” *arXiv preprint* arXiv:2505.03831, 2025.

[13] V. Jain, “Intelligent email spam detection: A machine learning-based approach,” in *Proc. 5th Int. Conf. Trends Mater. Sci. Invent. Mater. (ICTMIM)*, 2025, pp. 1574–1579.

[14] S. Khan, L. Han, G. Mudassir, B. Guehguih, and H. Ullah, “3c3r, an image encryption algorithm based on bbi, 2d-ca, and sm-dna,” *Entropy*, vol. 21, no. 11, p. 1075, 2019.

[15] K. A. Kumar and M. Sivakumar, “Enhancing the spam detection for social media using naive bayes classifier in comparison with random forest,” in *AIP Conf. Proc.*, vol. 3300, p. 020013, 2025.

[16] S. Mahalakshmi, D. L. Pansy, and V. M. Thejashree, “Smart email filtering against phishing attacks,” in *Proc. Int. Conf. Data Sci., Agents & Artif. Intell. (ICDSAAI)*, 2025, pp. 1–6.

[17] V. Metsis, I. Androutsopoulos, and G. Paliouras, “Spam filtering with naive bayes – which naive bayes?,” *CEAS*, 2006.

[18] A. Okunola and A. Ahsun, “Comparative analysis of machine learning models for real-time fraud detection,” *ResearchGate*, Jan. 2025.

[19] S. Prakash, B. Kalaiselvi, K. Sivachandar, *et al.*, “Recognizing fake documents by instance-based ML algorithm tuning with neighborhood size,” *J. Appl. Data Sci.*, vol. 6, no. 2, pp. 1214–1228, 2025.

[20] L. G. A. Putri, S. A. Wicaksono, and B. Rahayudi, “Analisis klasifikasi spam email menggunakan metode extreme gradient boosting (xgboost),” *J. Pengemb. Teknol. Inf. Ilmu Komput.*, vol. 9, no. 2, 2025.

[21] N. R. Rao and G. A. F. Vinodhini, “Measuring the efficiency of random forest, naive bayes, multilayer perceptron and support vector machine in email spam detection,” in *AIP Conf. Proc.*, vol. 3270, p. 020052, 2025.

[22] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, “A Bayesian approach to filtering junk e-mail,” in *Learning for Text Categorization: Papers from the 1998 Workshop*, vol. 62, pp. 98–105, Madison, WI, USA, 1998.

[23] M. Sharabov, G. Tsochev, V. Gancheva, and A. Tasheva, “Filtering and detection of real-time spam mail based on a Bayesian approach in university networks,” *Electronics*, vol. 13, no. 2, p. 374, 2024.

[24] K. Soppari, B. Vangapally, S. S. Sohail, and H. Dubba, “Survey on: Voice driven email solutions for visually impaired people,” *World J. Adv. Res. Rev.*, vol. 26, no. 1, pp. 032–036, 2025.

[25] E. H. Tusher, M. A. Ismail, and A. F. M. Raffei, “Email spam classification based on deep learning methods: A review,” *Iraqi J. Comput. Sci. Math.*, vol. 6, no. 1, p. 2, 2025.

[26] E. H. Tusher, M. A. Ismail, M. A. Rahman, A. H. Alenezi, and M. Uddin, “Email spam: A comprehensive review of optimize detection methods, challenges, and open research problems,” *IEEE Access*, 2024.

[27] H. Yin, W. Zhu, C. Fei, and X. He, “Deep learning for spatiotemporal modeling: A survey,” *IEEE Trans. Big Data*, 2017.

[28] T. Yin, W. Ding, H. Ju, J. Huang, and Y. Chen, “The fuzzy hypergraph neural network model based on sparse k-nearest neighborhood granules,” *Appl. Soft Comput.*, vol. 170, p. 112721, 2025.

[29] T. Yusnanto, F. Fatkhurrochman, M. A. Muin, and K. Mustofa, “Data security analysis on the use of e-commerce to prevent online fraud,” *RIGGS: J. Artif. Intell. Digit. Bus.*, vol. 4, no. 1, pp. 50–55, 2025.