

Improving Spam Detection Techniques Using Machine Learning Algorithms

Abdullah Said Alkabani^{a,*}, Iin Lesmana Putra^b and Dini Oktarina Dwi Handayani^c

^aAl Buraimi University, Oman

^bNusa Putra University, Indonesia

^cInternational Islamic University Malaysia, Malaysia

ARTICLE INFO

Keywords:

spam detection
XGBoost
F1-score
machine learning
email classification
support vector machine
random forest

ABSTRACT

Spam detection has become a critical challenge in maintaining the integrity and security of digital communication systems. This study focuses on enhancing spam detection precision by applying the XGBoost algorithm, compared to other machine learning models such as Support Vector Machines (SVM) and Random Forests. Traditional rule-based methods, while straightforward, need to address the increasing sophistication of spam tactics, leading to high rates of false positives and negatives. Machine learning, with its data-driven adaptability, provides a powerful alternative for improving spam detection accuracy. This research evaluates the effectiveness of XGBoost in handling large and complex datasets, leveraging its ability to build robust predictive models by combining weak learners. Performance metrics, including accuracy, precision, recall, and F1-score, are used to assess its efficacy. Additionally, a comparative analysis highlights XGBoost's advantages over SVM, which is known for its precision in high-dimensional spaces, and Random Forest, which is valued for its resilience against overfitting. The study emphasizes the importance of parameter optimization and feature engineering in maximizing the performance of spam detection models. This research contributes to developing more reliable and scalable spam detection frameworks by addressing the diverse attributes of spam messages. The findings offer valuable insights for researchers and practitioners in cybersecurity, underscoring the transformative potential of XGBoost in combating the evolving threat of spam.

1. Introduction

In today's digital age, spam emails have become a major threat to communication systems, accounting for over 50% of global email traffic, with some studies estimating it to be as high as 85% (McKinsey & Company, 2024). These unsolicited messages often serve as vectors for phishing, malware, and fraud, disrupting user productivity and posing significant security risks. The increasing sophistication of spam—employing social engineering tactics and advanced evasion methods—has rendered traditional rule-based detection systems less effective. Such systems are rigid, require constant manual updates, and frequently result in high false positive and false negative rates (Ozkan-Okay et al., 2023; Thakur et al., 2022).

To overcome these limitations, machine learning techniques have emerged as a promising alternative. Unlike static rule-based systems, machine learning models can learn patterns from large datasets and dynamically adapt to new forms

of spam. Among the widely used models in spam detection are Extreme Gradient Boosting (XGBoost), Support Vector Machines (SVM), and Random Forest. XGBoost is known for its speed, scalability, and robustness against overfitting, making it suitable for complex classification tasks like spam detection (Awad and Elseuofi, 2011; Gangavarapu et al., 2020). SVMs are effective in handling high-dimensional data and work by constructing optimal hyperplanes to separate classes (Awad and Elseuofi, 2011; Tamilarasan et al., 2023). Random Forest, an ensemble method that combines multiple decision trees, improves classification accuracy and mitigates overfitting (Gangavarapu et al., 2020; Gibson et al., 2020).

This study aims to evaluate and compare the performance of XGBoost, SVM, and Random Forest in detecting spam emails. Performance will be assessed using key evaluation metrics such as accuracy, precision, recall, and F1-score. Through this comparative analysis, the study seeks to identify the most effective machine learning approach to enhance the accuracy and efficiency of spam filtering systems, thereby contributing to the improvement of cybersecurity and digital communication reliability.

*Corresponding author

 iinlesmana.putra@nusaputra.ac.id (I.L. Putra);
dinihandayani@iium.ac.my (D.O.D. Handayani)
ORCID(s):

2. Research Methodology

2.1. Research Methodology Process

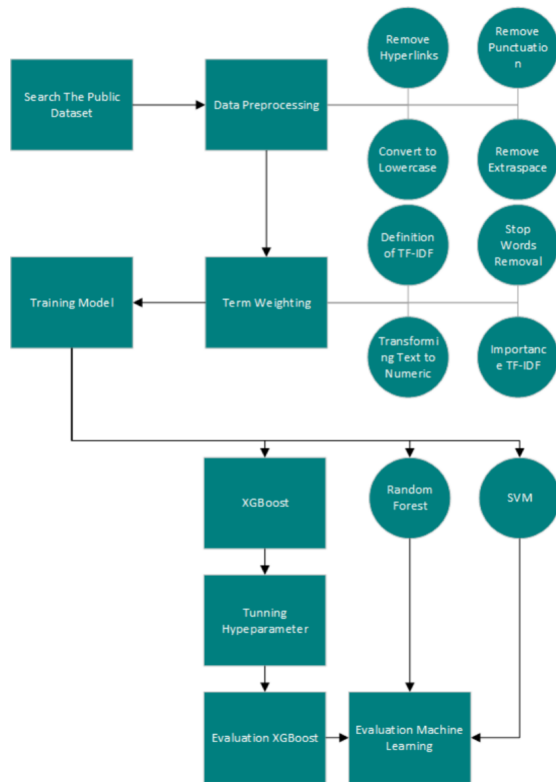


Figure 1: Research Methodology Process

The methodology illustrated in Figure 1 describes the research workflow from data collection to model evaluation. The process begins with searching for a publicly available dataset relevant to spam detection. The obtained dataset is then cleaned through a preprocessing stage using a Python script with regular expressions to remove hyperlinks, punctuation, inconsistent letter casing, excessive whitespace, and common stop words that do not contribute significantly to the meaning of the text. Once sanitized, the text is transformed into numerical representations using the TF-IDF (Term Frequency–Inverse Document Frequency) method, which assigns weights to each word based on its importance within the document. The processed dataset is subsequently divided into 80% training data and 20% testing data. Next, three machine learning algorithms—XGBoost, Random Forest, and SVM—are trained using the training data. For the XGBoost model, hyperparameter tuning is performed to achieve the optimal configuration and improve model performance. The final stage is evaluation, where the tuned XGBoost model’s performance is compared with Random Forest and SVM to obtain a model with high accuracy while minimizing both false positives and false negatives in spam detection.

2.2. Dataset Overview

The dataset used in this study was obtained from Kaggle, consisting of a labeled collection of spam and non-spam

emails. Although clean and well-structured, it does not fully capture real-world complexities such as noisy data, mislabeled samples, evolving spam tactics, and natural class imbalance. After verification and data cleansing, 83,448 valid records remained from the original 755,288 entries. The dataset contains two key columns: label (indicating spam = 1 or ham = 0) and text (the full email content). The text data, ranging from short messages to lengthy paragraphs, underwent natural language processing (NLP) to extract features for training and evaluating the classification models.

Table 1: The Example of Dataset

Label	Text
1	ounce feather bowl hummingbird opec moment alarm bells...
1	wulvob get your medications online qnb ikud viagra cialis...
0	computer connection from cnn com wednesday esteemed partner...
1	university degree obtain a prosperous future many career...
0	thanks for all your answers guys i know i should have...
0	larry king live at escape number escape number perry...
0	michael pobega wrote i'm not sure if it's the same...
0	hi i have this error tr sample escape number escape...
1	works gateway world art explore tattooing full report...
1	upon this account he is not only very cautious...

2.3. Model Training

This study employs the XGBoost (eXtreme Gradient Boosting) framework to build a robust phishing and spam detection model. XGBoost is an advanced implementation of gradient boosting that combines multiple weak learners, typically decision trees, to form a strong predictive model (Chen and Guestrin, 2016). Its key strengths include regularization (L1 and L2) to reduce overfitting, efficient handling of missing data, parallel and distributed computation for scalability, integrated pruning methods, and built-in cross-validation for streamlined tuning (Brownlee, 2016). Compared to deep learning models such as LSTM (Long Short-Term Memory) or Transformers, XGBoost offers high accuracy with lower computational cost and is well suited to the TF-IDF feature representation used in this study, which does not retain sequential dependencies (Goldberg, 2017). Class imbalance was addressed through the `scale_pos_weight` parameter, while other resampling techniques like SMOTE (Synthetic Minority Over-sampling Technique) or undersampling were not applied (Chawla et al., 2002). Beyond adjusting `n_estimators`, future fine-tuning could include parameters such as `max_depth` (to control tree complexity), `learning_rate` (to control the contribution of each tree), `subsample` (to prevent overfitting), and regularization parameters (`alpha` and `lambda`) to penalize model complexity. Though the current tuning focused on estimator count, the performance gain demonstrated the importance of this adjustment. Automating this process using grid search, randomized search, or Bayesian optimization could further optimize results, especially when dealing with evolving spam patterns in production systems.

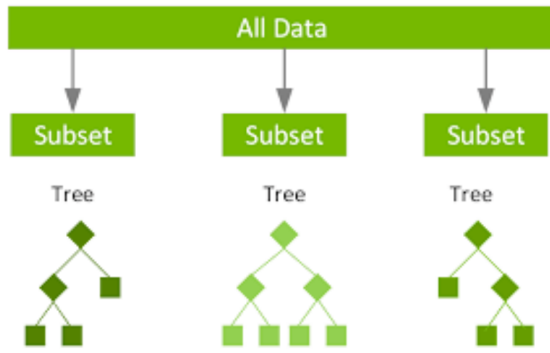


Figure 2: XGBoost Structure

2.4. Model Evaluation

The models were evaluated using conventional performance indicators, including accuracy, precision, recall, and F1 score. Cross-validation was conducted to verify their reliability and applicability. The efficacy of the integrated model was evaluated by comparing its performance with that of individual algorithms.

Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

Precision

$$\text{Precision} = \frac{TP}{TP + FP} \quad (2)$$

Recall

$$\text{Recall} = \frac{TP}{TP + FN} \quad (3)$$

F1 Score

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

Where:

TP = True Positives

TN = True Negatives

FP = False Positives

FN = False Negatives

3. Research Results

In this study, the initial stage involved a text preprocessing pipeline to prepare raw email data for subsequent model training. A total of 3,000 email data entries were used, which were split into two parts: 80% (2,400 emails) for training and 20% (600 emails) for testing. The preprocessing process included removing hyperlinks, punctuation marks, excessive spaces, and normalizing all characters to lowercase so that the text became cleaner and more consistent. After preprocessing, the cleaned text was converted

into numerical representations using the Term Frequency–Inverse Document Frequency (TF-IDF) method, resulting in relevant feature vectors for the machine learning algorithms. These processed features were then fed into the machine learning program for training and evaluation.

Based on the experimental results shown in the confusion matrix in Figure 3, the performance of the model in detecting spam and non-spam messages is illustrated. The evaluation reveals that the model correctly predicted 220 spam messages as spam and 323 non-spam messages as non-spam. However, there were still some misclassifications, with 53 spam messages incorrectly identified as non-spam and 4 non-spam messages misclassified as spam. These results demonstrate that the model performs very well in identifying non-spam messages, though improvements are still needed in terms of recall to more effectively capture all spam messages.

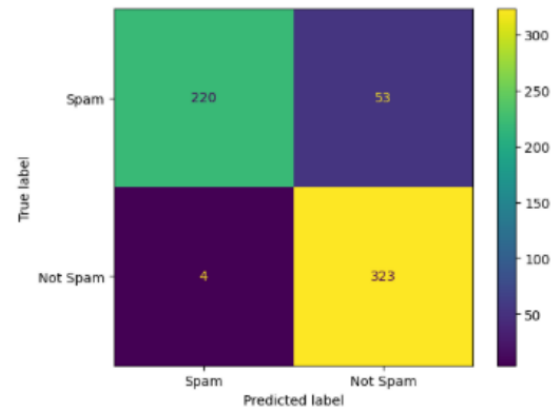


Figure 3: Classification Report for XGBoost Evaluation Model Before Tuning

The Random Forest model was evaluated using precision, recall, and F1-score metrics to assess its classification performance. As shown in Figure 4, the confusion matrix illustrates that the model correctly predicted 233 spam messages as spam (true positives) and 325 non-spam messages as non-spam (true negatives). However, 40 spam messages were misclassified as non-spam (false negatives), and only 2 non-spam messages were incorrectly classified as spam (false positives). These results indicate that the model demonstrates strong overall performance, particularly in accurately identifying non-spam messages, as reflected by the very low number of false positives. This high precision for the spam class shows that the model rarely mislabels legitimate emails as spam. Nevertheless, the presence of 40 false negatives suggests that the model still has room for improvement in recall to ensure that a higher proportion of spam messages is correctly detected.

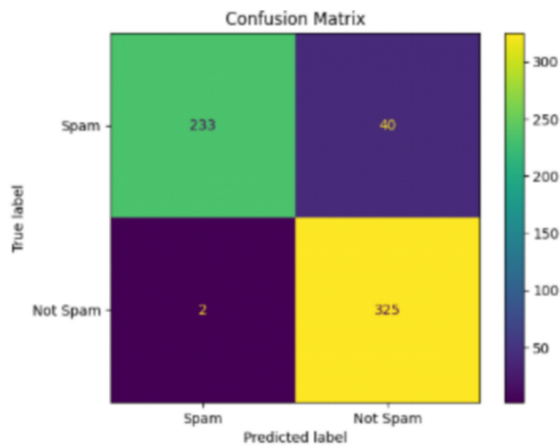


Figure 4: Classification Report for Evaluation Random Forest Model

The SVM model was evaluated using precision, recall, and F1-score metrics to measure its performance. As shown in Figure 5, the confusion matrix indicates that the model correctly predicted 225 spam messages as spam (true positives) and 326 non-spam messages as non-spam (true negatives). Only 1 non-spam message was incorrectly classified as spam (false positive), demonstrating that the model has very high precision, as it rarely mislabels legitimate messages as spam. However, 48 spam messages were incorrectly classified as non-spam (false negatives), indicating that the model's recall could be improved to better capture all spam messages.

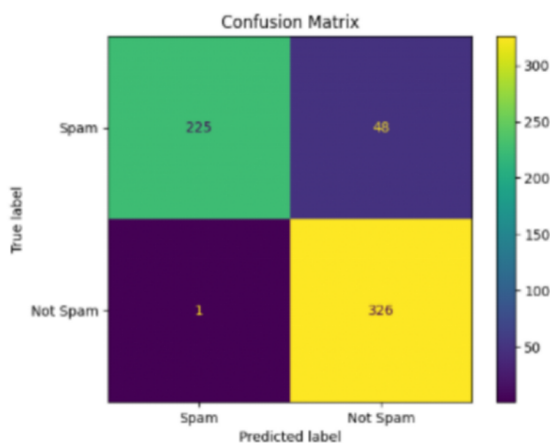


Figure 5: Confusion Matrix for SVM Evaluation Model

The XGBoost model, after tuning, produced the confusion matrix shown in Figure 6. In this evaluation, the x-axis represents the predicted labels and the y-axis represents the true labels for the two classes: "Spam" and "Not Spam." The results indicate that 252 spam messages were correctly classified as spam (true positives), while 319 non-spam messages were correctly classified as non-spam (true negatives). There were only 8 non-spam messages misclassified as spam (false positives) and 21 spam messages misclassified as non-spam (false negatives). These findings demonstrate that the tuned XGBoost model performs very effectively overall,

with a small number of misclassifications. The low false positive count shows that the model is careful in labeling messages as spam, which is important to avoid blocking legitimate emails. At the same time, the relatively low false negative count indicates that the model retains good sensitivity in detecting spam.

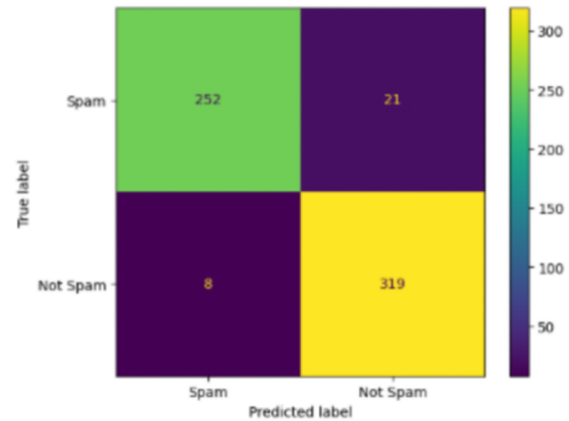


Figure 6: Confusion Matrix for XGBoost After Tuning Model

Figure 7 shows that, after tuning, the XGBoost model achieved the highest performance among all evaluated models, reaching an accuracy of 95.17% and an F1 score of 95.65%. In comparison, before tuning, XGBoost achieved a lower accuracy of 90.5% and an F1 score of 91.89%, demonstrating the significant improvement brought by parameter optimization. Random Forest ranked second with an accuracy of 93% and an F1 score of 93.93%, followed by the Support Vector Machine (SVM) with an accuracy of 91.83% and an F1 score of 91.72%. These results emphasize the critical role of hyperparameter tuning in enhancing XGBoost's performance, making it the most effective model for spam detection in this study. To illustrate these differences, a comparison graph of accuracy for each model was generated, as shown in Figure 7.

Among the three algorithms—XGBoost, Random Forest, and SVM—each demonstrated varying strengths. Initially, XGBoost's F1 score of 91.89% was respectable but slightly below Random Forest's 93.93%, while SVM followed closely with an F1 score of 91.72%. Random Forest performed well due to its ensemble approach, which balanced precision and recall by reducing overfitting through bootstrapped aggregation of decision trees. However, after fine-tuning hyperparameters such as the number of estimators (increased from 5 to 100), tree depth, and learning rate, XGBoost surpassed both models with an F1 score of 95.65%. This improvement demonstrates how careful parameter adjustments reduce bias and variance, enabling XGBoost to better capture complex patterns in spam messages. Furthermore, XGBoost's decision-making process can be explained through feature importance metrics and visualization tools like feature importance plots or decision path analysis, helping stakeholders understand the factors

influencing predictions and ensuring transparency and trust in practical deployment.

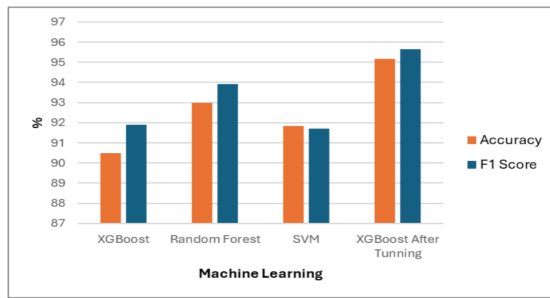


Figure 7: Comparison Graph of XGBoost, Random Forest, SVM, and Improved XGBoost

In addition to tuning, this research conducted experiments to evaluate how different parameter configurations in XGBoost affect performance. The two parameters modified were `n_estimators` (the number of boosting rounds or decision trees) and `learning_rate` (the contribution of each tree to the final prediction). Increasing the number of estimators allows the model to build more trees to correct previous errors, while adjusting the learning rate controls how aggressively the model learns from each iteration. A smaller learning rate enables more cautious learning and allows additional trees without overfitting, whereas a larger learning rate accelerates learning but risks instability if not paired with proper estimator counts.

As summarized in Table 2, a low number of estimators combined with a small learning rate (e.g., 100 estimators with 0.01 learning rate) led to underfitting and a lower accuracy of 83.83%. Accuracy improved as both parameters were optimized, such as 150 estimators with a 0.03 learning rate achieving 91.50%, or 5000 estimators with a 0.01 learning rate reaching 91%. The best results were observed with learning rates of 0.1 and estimator counts between 200 and 500, yielding accuracies up to 94.83%. Even with very high estimator counts (e.g., 20,000) and a moderate learning rate, accuracy plateaued at 94.83%, showing diminishing returns. Overall, these experiments demonstrate that a balanced combination—such as a moderately high number of estimators with a learning rate of 0.05 or 0.1—delivers the most effective spam classification performance, consistently achieving accuracies in the 94–95% range.

Table 2: Accuracy Comparison with Different Parameters

n_estimators	Learning Rate	Accuracy
100	0.01	83.83%
150	0.03	91.50%
5000	0.01	91%
200	0.1	94.50%
300	0.05	94%
10000	0.001	94.67%
20000	0.01	94.83%
2000	0.05	94.17%
500	0.1	94.83%

Figure 8 presents a SHAP (SHapley Additive exPlanations) summary plot that visualizes the 20 most influential

features affecting the predictions of the tuned XGBoost model. On the horizontal axis are the SHAP values, which represent how much each feature pushes the prediction toward Spam (positive values) or Not Spam (negative values), while the vertical axis lists the feature names. Each dot represents a single data instance, with its color indicating the feature's value: red denotes a high value or frequent occurrence of the feature, while blue indicates a low value or rare occurrence.

The plot reveals that certain words such as “org,” “enron,” and “wrote” have strong negative contributions to spam classification, especially when their frequencies are high. This suggests that emails containing these terms are more likely to be legitimate, which aligns with their context in corporate or organizational communication within the Enron dataset. In contrast, words such as “http,” “listinfo,” “unsubscribe,” “money,” and “project” show strong positive contributions to spam predictions when their values are high. These terms are commonly associated with spam-like characteristics, including promotional content, hyperlinks, subscription links, and financial offers. Additional words such as “save,” “online,” and “pm” also contribute positively to spam detection, though with a lower magnitude of impact.

These findings indicate that the tuned XGBoost model has successfully captured meaningful linguistic patterns that differentiate spam from non-spam emails. The improvement achieved through parameter tuning not only increased accuracy but also preserved interpretability, as the model's decisions are based on semantically explainable features. This level of transparency is important for building trust in real-world applications, allowing system developers and users to understand why certain messages are classified as spam or not spam, and ensuring the system remains auditable and adaptable over time.

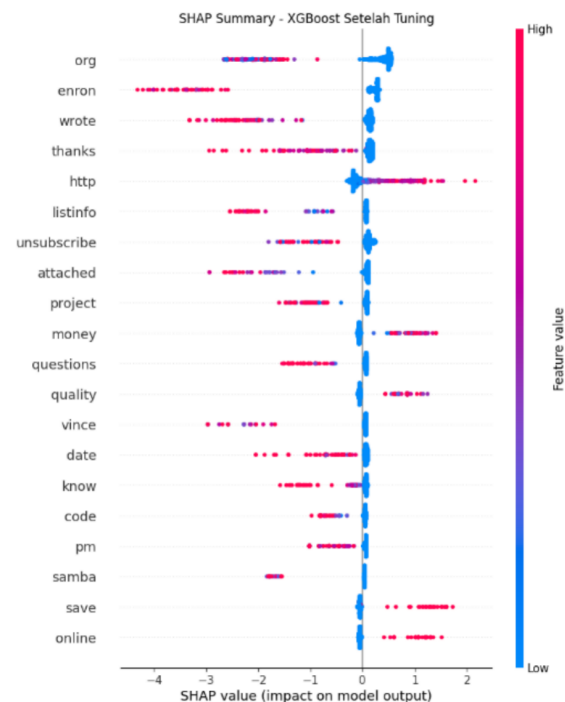


Figure 8: SHAP Summary the XGBoost Model After Tuning

4. Conclusions and Recommendations

4.1. Conclusions

This study comprehensively evaluates the performance of three machine learning algorithms—XGBoost, Random Forest, and Support Vector Machine (SVM)—in detecting spam messages, using evaluation metrics such as accuracy, precision, recall, and F1-score. The research emphasizes the importance of proper data preprocessing and feature engineering, where techniques like data cleansing and Term Frequency–Inverse Document Frequency (TF-IDF) vectorization were applied to transform raw email text into structured numerical features. These steps ensure that the models can learn effectively from the dataset, which in this study consisted of 3,000 email samples, with 80% used for training and 20% for testing.

Through systematic experimentation, it was found that XGBoost, after fine-tuning its parameters—particularly `n_estimators`—achieved the highest performance with an accuracy of 95.17% and an F1-score of 95.65%. Random Forest followed with strong but slightly lower results, and SVM performed reliably but lagged behind the other two. Parameter tuning was shown to play a critical role in improving classification performance, as demonstrated by the significant performance gain in XGBoost after optimization. Overall, the study successfully met its objectives by implementing, comparing, and optimizing these models, proving that well-tuned machine learning approaches can significantly outperform traditional rule-based spam detection systems in terms of scalability, reliability, and adaptability to evolving spam characteristics.

4.2. Recommendations

Subsequent research may expand upon the findings of this study by investigating the following avenues:

1. Future research can explore the integration of XGBoost with deep learning models, such as LSTMs or Transformers, to further enhance spam detection performance. This hybrid ensemble approach would leverage the complementary strengths of both model types—XGBoost’s capability to handle structured and tabular data efficiently, and deep learning’s ability to capture complex semantic and sequential relationships in unstructured text.
2. Integrating the model into a streaming architecture using tools like Apache Kafka or Apache Flink can enable real-time data ingestion and processing. This would allow the model to make predictions on incoming emails with minimal delay.
3. The XGBoost-based spam detection model can be integrated into email filtering systems, such as Gmail or Outlook. Still, the integration depends on access to the backend infrastructure and compliance with privacy protocols. Deploying the model on large-scale email servers introduces several significant challenges. One

primary concern is latency and throughput, as spam detection systems on platforms like Gmail must classify millions of emails per second. To meet such demands, the model must offer low inference time and support high-throughput deployment, which may require optimization techniques such as model quantization, batching, and parallelization. Scalability is another critical factor, where the system must scale horizontally to handle increasing email volumes without compromising accuracy or response time. This often involves deploying the model using containerization and orchestration tools like Docker and Kubernetes, which enhance scalability but add to the deployment complexity. Security and privacy concerns also play a pivotal role, as email content frequently contains sensitive personal or organizational data. Therefore, any deployed solution must comply with data protection regulations such as the GDPR and may require anonymization techniques or on-device inference to ensure data confidentiality. Furthermore, because spam tactics evolve continuously, the model must be capable of continuous learning. This requires regular updates through retraining on new data or implementing automated learning pipelines that enable the system to adapt to new threats in real-time. Lastly, false positives—where legitimate emails are incorrectly flagged as spam—can significantly affect user experience and trust. To minimize this risk, incorporating threshold calibration, enhanced explainability, and user feedback loops into the system is recommended.

References

- Awad, W. A. and Elseuofi, S. M. (2011). Machine learning methods for spam email classification. *International Journal of Computer Science and Information Security*, 9(9):360–364.
- Brownlee, J. (2016). A gentle introduction to XGBoost for applied machine learning. *Machine Learning Mastery*.
- Chawla, N. V. et al. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.
- Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794.
- Gangavarapu, S. et al. (2020). Spam detection using ensemble learning methods. *Procedia Computer Science*, 167:2201–2210.
- Gibson, D. et al. (2020). A comparative study of machine learning algorithms for spam detection. *Journal of Information Security*, 11(3):131–140.
- Goldberg, Y. (2017). Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 10(1):1–309.
- McKinsey & Company (2024). Cybersecurity and digital threats: Global trends and risks. <https://www.mckinsey.com>.
- Ozkan-Okay, A., Yilmaz, E., and Akin, E. (2023). Rule-based and learning-based methods for spam detection: Challenges and future directions. *Expert Systems with Applications*, 222:119–145.
- Tamilarasan, R. et al. (2023). Comparative analysis of classification algorithms in spam email detection. *International Journal of Advanced Computer Science and Applications*, 14(2):123–130.
- Thakur, R., Joshi, R., Thakral, R., and Jain, A. (2022). Detection of spam emails using machine learning algorithms. *Materials Today: Proceedings*, 56:409–415.