Improving Spam Detection with Feature Engineering and Adaptive Learning Approaches

Sadeem H. AlHomidan, Marwah M. Almasri, Shimaa A. Nagro
College of Computing and Informatics, Saudi Electronic University, Riyadh, Saudi Arabia

***Abstract*—Spam email detection is a critical component of securing and maintaining reliable digital communication systems. This study explores the effectiveness of various machine learning algorithms in classifying spam, with an emphasis on enhancing accuracy and precision through systematic preprocessing, advanced feature engineering, and text preprocessing. Six models were evaluated: Logistic Regression, Support Vector Classifier, Multinomial Naïve Bayes, K-Nearest Neighbors, AdaBoost, and Bagging Classifier using a comprehensive preprocessing pipeline that included Term Frequency–Inverse Document Frequency vectorization, feature scaling, and the incorporation of engineered features such as character counts. Experimental results reveal that Multinomial Naïve Bayes consistently achieved the highest precision 1.00 and strong accuracy 0.979 when paired with feature scaling, while Logistic Regression delivered robust and stable performance across multiple configurations with precision exceeding 0.96, making it a reliable choice for real-world deployment. Although Support Vector Classifier and AdaBoost exhibited competitive baseline performance, Support Vector Classifier showed limitations when handling numeric features, whereas AdaBoost maintained consistent results across scenarios. These findings underscore the critical role of tailored preprocessing and ensemble learning in improving classification outcomes and highlight the comparative strengths of different algorithms in real-world spam detection. In particular, Multinomial Naïve Bayes proved highly effective for precision-critical tasks, while Logistic Regression emerged as a dependable solution for environments requiring consistent reliability. Overall, this work advances machine learning-based spam filtering by identifying models that successfully balance precision, adaptability, and computational efficiency.**

Keywords*—Spam detection; machine learning; Multinomial Naïve Bayes; logistic regression; ensemble learning; text preprocessing; feature engineering

I. Introduction

Email remains one of the most widely used communication tools in both personal and professional contexts due to its speed, efficiency, and ability to handle large volumes of data. However, the exponential growth of email usage has been accompanied by an alarming rise in spam, unsolicited messages often used for phishing, malware distribution, and other cybercrimes. Spam not only clutters inboxes and reduces productivity but also undermines trust in email systems and introduces severe cybersecurity risks.

Early detection approaches, such as manual filtering and keyword-based rules, quickly became ineffective as spammers adopted evasion techniques like word obfuscation (e.g., “fr33” instead of “free”), embedding text in images, or mimicking legitimate content. These limitations underscored the need for adaptive, intelligent methods. Machine learning (ML) has since emerged as a powerful solution, as models can learn patterns from large labeled datasets and adapt to evolving spam strategies. Combined with natural language processing, ML enables the capture of deeper semantic patterns beyond simple keyword matching [1], [2]. Nevertheless, challenges persist, including scalability with massive email volumes, adaptability to new spam tactics, and the risks of false positives and false negatives, which either disrupt legitimate communication or expose users to malicious content [3].

Despite the progress in ML-based spam detection, several limitations remain. Conventional filters fail against modern spam tactics like obfuscation, ambiguous phrasing, or image-based content [4], [5]. While deep learning models such as BERT or GPT-based architectures represent the current state-of-the-art in text understanding, their deployment often requires substantial computational resources and large-scale training data, which may limit their practicality for lightweight or real-time spam filtering environments. In contrast, classical ML approaches remain highly competitive for short-text classification tasks—such as spam detection—especially when applied to small and medium-sized datasets, where deep learning models tend to overfit. Moreover, these lightweight models offer faster inference, lower memory requirements, and easier integration into existing email systems. Therefore, this study focuses on efficient, traditional ML models that balance accuracy, scalability, and real-world deployability without the computational overhead associated with deep learning.

This study develops and evaluates six ML models—Logistic Regression (LR), Support Vector Classifier (SVC), Multinomial Naive Bayes (MNB), K-Nearest Neighbors (KNN), AdaBoost, and Bagging—to improve spam detection performance over traditional methods. The contributions of this study are threefold:

Advancing spam detection research by comparing classical ML and ensemble methods, highlighting their strengths and weaknesses under different spam scenarios.

Demonstrating the impact of systematic preprocessing and feature engineering on improving classification outcomes and reducing errors.

Providing practical insights for real-world deployment, identifying models that combine accuracy, stability, and computational efficiency, thereby enhancing email system security and user trust.

The remainder of this study is organized as follows: Section II reviews related literature on spam detection and ML. Section III presents the dataset and describes the preprocessing and feature engineering techniques applied. Section IV outlines the methodology and experimental setup. Section V discusses the results and their implications. Finally, Section VI concludes the study and highlights directions for future work.

II. Literature Review

The evolution of spam detection has undergone a significant transformation over the past two decades. The literature is highlighting a clear progression from manual filtering approaches to advanced hybrid frameworks that integrate classical ML, ensemble learning, and contextual embeddings.

The earliest spam filtering solutions relied heavily on manually defined rules, blacklists, and simple keyword-matching techniques. These systems flagged messages based on predefined patterns or suspicious terms such as "win", "free", or "offer", often combined with header and metadata checks. While effective in early scenarios, such methods lacked adaptability. Attackers quickly learned to bypass these filters by using obfuscation, intentional misspellings, and paraphrased language. As a result, the limitations of manual and keyword-based approaches became evident, particularly in their inability to handle evolving spam tactics and large-scale data.

To overcome the rigidity of manual filters, the field shifted toward data-driven models capable of learning discriminative patterns directly from labeled datasets. Classical ML techniques, including MNB, LR, SVMs, Decision Trees, and KNN, became central to spam detection research [6]. Among them, LR has remained a fundamental baseline due to its probabilistic interpretation, simplicity, efficiency in handling high-dimensional text data, and competitive performance when paired with feature engineering techniques such as Term Frequency–Inverse Document Frequency (TF-IDF), n-grams, and regularization [7]. The study has demonstrated LR's effectiveness even against more complex models. For example, enhancements to LR through feature selection methods (e.g., chi-square, PCA, or recursive feature elimination) and optimized preprocessing pipelines have led to high accuracy and robustness on email and SMS spam datasets.

Despite their strong baseline performance, classical models often struggled with imbalanced datasets, noisy inputs, and non-linear decision boundaries. This led to the rise of ensemble techniques, such as bagging, boosting, and stacking, which combine the strengths of multiple classifiers to improve generalization and robustness. Recent research highlights the use of LR as a meta-learner in stacking frameworks, where predictions from base learners like NB, SVM, DT, and KNN are fed into LR to optimize final classification. These ensemble approaches consistently outperform individual models, achieving accuracy improvements of up to 5–10% over traditional baselines [8].

With the advent of deep learning, spam detection systems began leveraging neural networks capable of learning hierarchical text representations without manual feature engineering. Convolutional Neural Networks and Recurrent Neural Networks initially dominated the field [9], but Transformer-based architectures such as BERT, RoBERTa, and DistilBERT have since become state-of-the-art due to their ability to model contextual semantics and long-range dependencies. Comparative studies show that Transformer-based models significantly outperform traditional ML approaches, particularly in handling obfuscated, paraphrased, or multilingual spam.

Recent work addresses the growing threat of SMS spam—responsible for credential theft and data loss—by combining GPT-3-based text embeddings with ensemble learning for improved classification. This hybrid model significantly outperforms individual classifiers, achieving 99.91% accuracy on the SMS Spam Collection dataset. Such results highlight the effectiveness of integrating Transformer representations with ensemble techniques in modern spam detection [10].

Overall, the literature demonstrates a clear progression from manual filters to keyword-based systems, then to classical ML models, and finally toward ensembles and deep learning architectures. Current research emphasizes the importance of hybrid frameworks that combine classical ML, ensemble learning, and contextual embeddings to achieve both accuracy and adaptability. Nevertheless, balancing detection performance with scalability, robustness, and interpretability remains an ongoing challenge for real-world deployment of spam detection systems.

III. Dataset and Preprocessing

The primary dataset used in this study is the SMS Spam Collection Dataset, comprising 5,574 email messages labeled as either "spam" or "ham" (legitimate). The dataset is organized into two columns:

v1 – the label indicating whether the message is *spam* or *ham*.

v2 – the raw text content of the email message.

As indicated in [11], the spam subset includes 425 messages collected from the Grumble Text website, a UK-based forum where users report spam messages they have received. Since these reports often contained a mix of relevant and irrelevant text, the spam content had to be carefully extracted through extensive manual filtering.

The ham (legitimate) subset primarily comprises 3,375 messages sourced from the NUS Corpus, which was compiled from emails provided by students at the National University of Singapore for research purposes. An additional 450 ham messages were obtained from Caroline Tag's PhD thesis to further enrich the dataset.

Extended Corpus: To enhance dataset diversity and improve model generalization, the Spam Corpus v.0.1 Big was also integrated, contributing an additional 1,002 ham messages and 322 spam messages. This extended corpus has been widely used in prior academic research on spam detection.

This dataset was chosen for its diversity and representativeness, combining global and regional spam and ham messages. Such heterogeneity makes it particularly well-suited for training a robust and generalizable spam detection

system with strong coverage across various email types and sources.

A. Data Cleaning and Dimensionality Reduction

Since the primary objective of this study is to detect spam emails, the focus is placed only on features directly relevant to the classification task, such as the email content and subject line. Some features that can be extracted from raw emails, like the timestamp, unique server ID, routing information, or execution data from the server, do not contribute meaningfully to determining whether a message is spam. Including such irrelevant features may introduce noise, reduce model interpretability, and negatively affect performance.

Therefore, as part of the dimensionality reduction process, unnecessary columns were removed, as they provided no analytical value. This approach ensures that the employed model captures only the most informative features, thereby improving training efficiency and reducing the risk of overfitting.

Another essential part of data cleaning is addressing data redundancy. Duplicate records, such as emails with identical content, similar subjects, or those sent repeatedly from the same sender, can bias the model, leading to skewed results and overfitting.

Duplicate records are identified by comparing the similarity of message content, metadata fields, and header information. Emails that share substantial similarities or are exact replicas of others are flagged as duplicates. Once identified, duplicate emails are removed from the dataset to ensure that each data point contributes unique information. This step reduces bias, minimizes noise, and enhances the overall robustness and generalization capability of the spam detection model.

By eliminating irrelevant features and removing duplicates, the dataset becomes cleaner, more concise, and better structured, ultimately improving the model's performance and its ability to classify spam emails accurately. After this preprocessing step, the final dataset comprises 5,169 email messages.

B. Exploratory Data Analysis

The dataset contains 5,169 emails, including 4,516 ham (87%) and 653 spam (13%) messages, showing a clear class imbalance. This imbalance can significantly affect model performance, as a naïve classifier might predict only "ham" to achieve high accuracy, leading to poor spam detection. To address this issue, class weighting was applied during model training to give higher importance to the minority class (spam) and improve the detection performance.

While synthetic oversampling techniques such as SMOTE can be used to mitigate class imbalance, they were intentionally not applied in this study. Because the dataset consists of short text messages, generating synthetic samples may distort the natural linguistic distribution of spam content and introduce patterns that do not occur in real data. Moreover, oversampling can lead to overfitting, especially in small or medium-sized datasets such as this one. Therefore, class weighting was preferred as a more reliable approach that preserves the authenticity of the original messages while still improving minority-class detection.

Text length–based features, including character count, word count, and sentence count, provide valuable indicators for distinguishing spam from ham messages. Spam messages are generally longer, averaging around 137 characters compared to 70 characters for ham. A similar pattern is seen with word count, where spam messages contain about 28 words on average, while ham messages contain around 17. Spam also tends to include more sentences (about 3) compared to ham (1–2), reflecting their more detailed or promotional nature. Fig. 1 presents histograms of character and word counts, showing distinct distribution patterns, with ham messages generally exhibiting higher counts across features, while spam messages are more concentrated in lower ranges. This separation indicates that these features carry valuable discriminative information, making them effective indicators for distinguishing between spam and ham emails. Moreover, character count, word count, and sentence count are strongly positively correlated, as one increases, so do the others. While this interdependence enhances classification performance, it also introduces potential redundancy, suggesting the need for dimensionality reduction or selective feature use. Overall, these text length features significantly improve the model's ability to differentiate spam from legitimate messages.

C. Data Preprocessing

To prepare the email text for feature extraction and model training, several preprocessing steps were applied to transform the raw content into a clean, standardized format. These steps include:

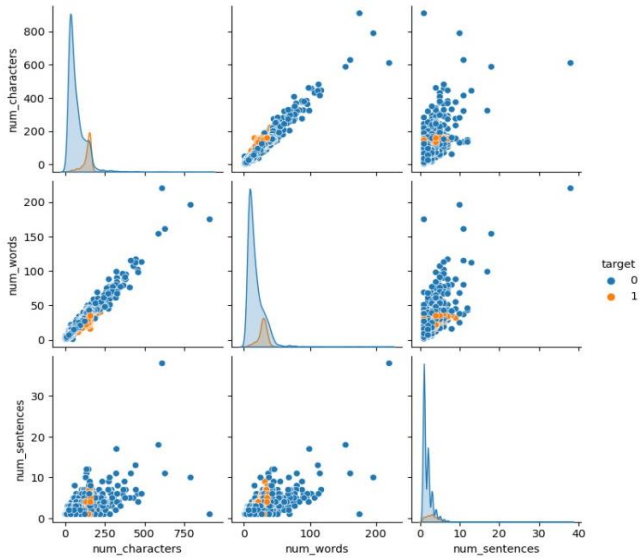

Fig. 1. Pairwise histograms of key textual features—character count, word count, and sentence count—for spam (1) and ham (0) messages. The diagonal panels show the distribution of each feature, while the off-diagonal scatter plots illustrate correlations between feature pairs, highlighting clear structural differences between spam and ham messages.

1) Lowercasing: All text was converted to lowercase to ensure uniformity (e.g., treating "Spam" and "spam" as the same word).

2) Tokenization: Text was split into individual tokens (words) for further analysis.

3) Removing special characters: Non-alphanumeric symbols, such as punctuation and special characters, were removed to reduce noise and focus on meaningful content.

4) Removing stop words: Common, less informative words (e.g., "the", "is", "and") were filtered out using a predefined English stop-word list.

5) Stemming: Words were reduced to their root form using the Porter Stemmer (e.g., "crying" → "cri", "loving" → "love") to group similar terms and minimize vocabulary size.

Text Vectorization and feature extraction: Text data must be converted into numerical form before it can be processed by ML models. Two commonly used techniques for this purpose are Bag of Words and TF-IDF. In this study, the TF-IDF vectorizer was used with a maximum of 3000 features to capture the most significant terms in the dataset. This transformation converted the cleaned email text into a numerical feature matrix with a shape of 5169, 3000, where 5169 represents the number of email samples and 3000 the most frequent terms considered. Limiting the vocabulary size reduces computational cost, mitigates overfitting, and ensures the model focuses on the most relevant features.

In addition to text-based features, three structural features were extracted to further enhance the model's predictive performance: number of characters, number of words, and number of sentences in each email. These features capture length and structural patterns that often differ between spam and ham messages, for example, spam messages typically contain more characters, words, and sentences. Combining these engineered features with TF-IDF representations strengthens the model's ability to accurately distinguish spam from legitimate emails.

Fig. 2 visualizes the most commonly occurring words in spam and ham messages. Spam emails (left) prominently feature marketing-related terms such as "free", "text", "win", and "claim", reflecting their promotional and persuasive intent. In contrast, ham messages (right) contain conversational words like "got", "come", "love", and "know", highlighting their informal and context-driven nature. The distinct lexical patterns shown here demonstrate how vocabulary distribution can help distinguish spam from legitimate messages.

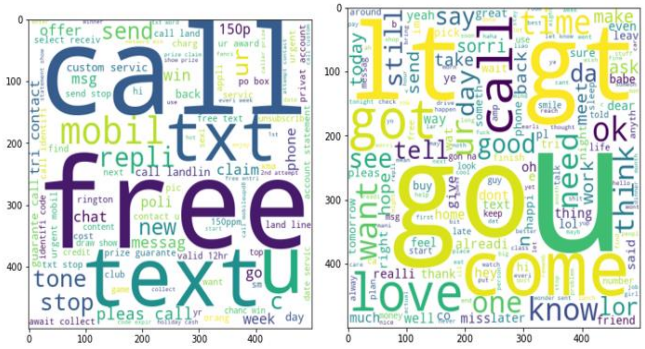

Fig. 2. Word clouds showing the most frequent terms in spam (left) and ham (right) messages. The spam cloud is dominated by promotional and action-oriented words such as "free", "call", "text", and "mobile", while the ham cloud features conversational terms are like "got", "you", "come", and "love". These contrasts highlight the distinct linguistic patterns that separate unsolicited messages from normal user communication.

IV. Methodology

Fig. 3 illustrates the complete workflow of the proposed spam detection system, beginning with the collection of email/SMS datasets. The data undergoes several preprocessing steps, including lowercasing, tokenization, removal of special characters and stop words, and stemming, to ensure text consistency and quality. Following preprocessing, various machine learning algorithms including LR, SVC, MNB, DBM, AdaBoost, and Bagging Classifier are initially trained and evaluated. Based on the evaluation results, feature extraction and engineering are then performed using the TF-IDF technique along with additional statistical features such as the number of characters, words, sentences, and links, with the vocabulary size set to 3000 features. Finally, the models are retrained and re-evaluated using the enhanced feature set to assess performance improvements.

A. Models' Building

Several models were trained on the training data using their default parameters, initially. After obtaining baseline results, models were fine-tuned to improve performance.

Logistic Regression is a Linear Model which predicts Probability of email being Spam given Input features. The decision boundary was found by applying this point classifier linear model to separate the two classes and give an interpretable and straightforward process for binary classification tasks.

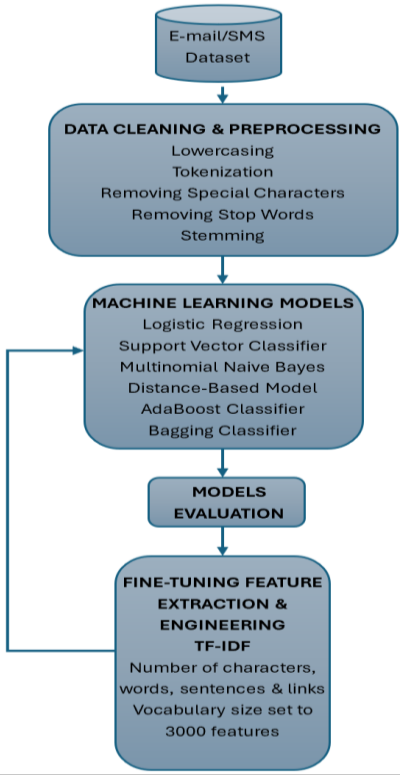

Fig. 3. Workflow of the proposed spam detection pipeline, illustrating the main stages from data collection and preprocessing to feature extraction, model training, and evaluation.

Support Vector Classifier (Tree Based Model) tries to classify between spam and ham email by finding the optimal hyperplane, essentially a high dimensional plane which gives us

the greatest separation for the classes. This model is good at discovering data distributions and works well for high dimensional data like text.

Multinomial Naive Bayes (Bayesian Model) is also applied, where the probability distribution of words in spam and ham emails is utilized. Owing to its simplicity, effectiveness, and ability to operate under the independence assumption between words, this model has become one of the standard approaches for text classification.

The Distance Based Model K-Nearest Neighbors classifies emails by the labels of the closest training samples. This is a non-parametric classification approach where decisions are made on the 'neighbors' of each email.

AdaBoost Classifier (Ensemble Method) comprises a combination of several weak classifiers like decision trees or linear models to make strong classifier. The improvement focus of this method is accuracy (addressing more emails than 80% of the emails are correctly classified).

And at last, the Bagging Classifier (Ensemble Method) trains a pool of models using a portion of the data, and then the number of predictions is summed up. The reason why this reduces overfitting is that, the model is able to generalize more and does well in new data.

The goal was to improve the accuracy and speed of this task through these diverse models by at least 80%.

B. *Models' Fine Tuning*

To enhance the performance of spam detection models, several new features were engineered to capture additional patterns:

1) Number of characters: Spam messages often have longer content, so the total character count was added as a numeric feature.

2) Number of links: Spam emails typically include multiple hyperlinks; counting them provides another useful indicator.

3) Presence of specific keywords: Binary features were created to capture the occurrence of common spam-related terms (e.g., "free", "win", "click here").

These engineered features enable the models to learn more complex patterns and improve classification accuracy.

Model evaluation was parameterized to compare the performance of various algorithms under different feature configurations, including limiting vocabulary size to 3000 features, applying data scaling, and incorporating character count. For each configuration, a DataFrame was generated to record metrics such as accuracy and precision, which were then compared across models. The results were merged into a single table to enable side-by-side performance comparison.

V. Results and Discussion

Model performance was primarily evaluated using accuracy, which measures the percentage of correctly classified spam and ham emails. However, given the class imbalance in the dataset, accuracy alone does not fully capture model effectiveness. Therefore, precision was also examined to assess how many emails predicted as spam were actually spam, a critical metric, since misclassifying legitimate emails as spam can significantly impact user experience.

Additionally, a confusion matrix was generated for NMB to analyze true positives, true negatives, false positives, and false negatives, providing deeper insight into classification errors and their nature. By comparing models across these metrics, the study aimed to identify a solution that balances performance, complexity, and practicality, ensuring an effective and reliable spam detection system.

Logistic Regression achieved strong baseline results with accuracy = 0.952 and precision = 0.940. Reducing TF-IDF features to 3000 improved both metrics (accuracy = 0.956, precision = 0.970) by focusing on the most informative terms. Feature scaling further boosted accuracy to 0.967 and slightly increased precision (0.964). Adding the *number of characters* feature slightly enhanced precision (0.971) without changing accuracy significantly. Overall, LR benefited most from feature reduction and additional feature engineering.

Support Vector Classifier performed strongly from the start (accuracy = 0.973, precision = 0.974), with only a slight improvement after feature reduction (accuracy = 0.975). Scaling slightly decreased precision (0.943) without affecting accuracy. Including the number of characters feature significantly reduced performance (precision dropped to 0.000), indicating that this feature is unsuitable for SVC. The best configuration was achieved with feature reduction and no additional features.

Multinomial Naive Bayes showed excellent results, achieving accuracy = 0.959 and a perfect precision = 1.000 at baseline. Feature reduction improved accuracy further (0.972) while maintaining perfect precision. Scaling raised accuracy to 0.979, though precision decreased slightly (0.946). Adding the number of characters feature reduced accuracy (0.940) but kept precision at 1.000. The confusion matrix, Fig. 4 confirmed its strength in classifying ham (TN = 896, FP = 0) but showed some false negatives (FN = 29). Overall, MNB offered the best trade-off between simplicity, precision, and performance.

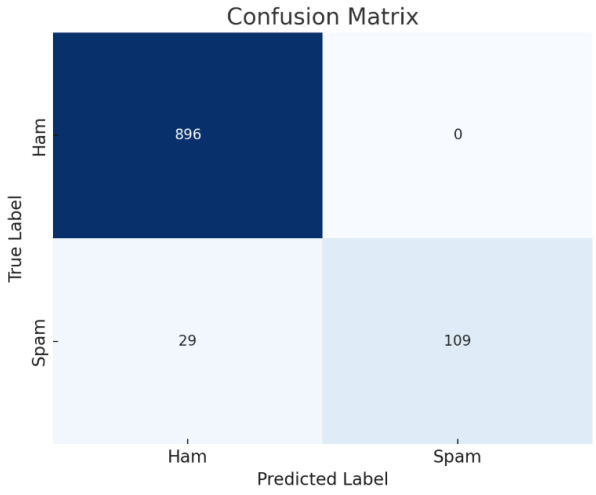

Fig. 4. Confusion matrix of the Multinomial Naive Bayes spam detection model, showing the distribution of correct and incorrect predictions across ham and spam classes.

This visualization clearly shows strong ham classification performance and highlights the small number of spam messages that were missed by the model.

K-Nearest Neighbors had a lower baseline accuracy (0.900) but perfect precision (1.000). Feature reduction slightly improved accuracy (0.905) while maintaining precision. Scaling further enhanced both metrics (accuracy = 0.905, precision = 0.976), reflecting the importance of normalization in distance-based models. Adding the number of characters feature improved accuracy (0.928) but reduced precision (0.771), suggesting added noise. The model performed best with feature reduction and scaling, but without additional features.

AdaBoost Classifier: AdaBoost delivered an accuracy = 0.962 and a precision = 0.954 at baseline. Feature reduction caused slight drops in performance (accuracy = 0.961, precision = 0.946), and scaling had minimal effect. Adding the number of characters feature enhanced both metrics (accuracy = 0.972, precision = 0.950), showing the feature's positive impact on ensemble performance.

Bagging Classifier: Bagging showed small improvements with feature reduction (accuracy = 0.959, precision = 0.869) and no significant changes after scaling. However, incorporating the number of characters feature substantially boosted performance (accuracy = 0.968, precision = 0.913), highlighting the feature's importance for this ensemble method.

The results presented in Table I demonstrate notable variations in model performance across different experimental settings. Overall, the SVC consistently achieved high performance, recording the highest accuracy of 0.975 and precision of 0.975 when the feature set was expanded to 3000, indicating its strong generalization capability with enriched feature representations. MNB achieved the highest overall accuracy of 0.979 under feature scaling, while KNN exhibited the best precision of 0.976 in the same setting, highlighting the sensitivity of these models to feature normalization. Furthermore, ensemble methods, particularly AdaBoost (0.972) and Bagging Classifier (0.968), outperformed others in the character-based feature setting, reflecting their robustness in handling feature diversity. LR also delivered competitive results, especially with a precision of 0.971 in the character-level configuration, but no single model dominated across all scenarios. These results support the hypothesis that the number of character features is particularly beneficial to ensemble methods such as Bagging and AdaBoost, whereas feature reduction (max feature) significantly enhances the performance of Naive Bayes and SVC models by improving their ability to capture essential features while reducing noise. Overall, these findings underscore the critical influence of feature engineering and preprocessing on model performance and suggest that the optimal choice of algorithm depends on the specific task requirements and prioritized evaluation metrics.

A significant class imbalance was observed in the dataset, with the number of ham messages greatly exceeding the number of spam messages. This imbalance posed a substantial challenge for the classifiers, as they tended to focus disproportionately on the majority class, making it difficult to accurately detect spam. As a result, performance metrics such as precision and accuracy can be somewhat misleading, as they may reflect high performance on ham messages while failing to capture deficiencies in spam detection. The impact of this imbalance varied across models. LR achieved 95.2% accuracy and 94.0% precision initially, but incorporating the number of characters feature improved its spam detection capability, demonstrating its capacity to learn from feature enhancements. SVC showed strong baseline performance (97.3% accuracy, 97.4% precision), but precision dropped to 0.0 when the number of characters feature was added, indicating a severe bias toward ham messages. MNB performed well, achieving perfect precision (1.0) and demonstrating robustness to imbalance, though it could still fail to identify some spam messages. KNN achieved 90% accuracy and perfect precision at baseline, but struggled with spam detection due to the skewed distribution. AdaBoost reached 96.2% accuracy and 95.4% precision but remained sensitive to class imbalance, excelling at classifying ham while struggling with spam. Bagging Classifier achieved 95.7% accuracy but only 86.2% precision, and even with resampling and the inclusion of a number of character features, spam detection remained challenging. Considering accuracy, interpretability, and robustness to imbalance, LR, SVC, and MNB emerge as the most suitable models. In real-world applications, these models could be deployed individually or combined into an ensemble to further enhance spam detection performance.

TABLE I. PERFORMANCE COMPARISON OF SIX MACHINE LEARNING MODELS

Models		***LR***	***SVC***	***MNB***	***KNN***	***AdaBoost Classifier***	***Bagging Classifier***
Base line	accuracy	0.952	0.973	0.959	0.900	0.962	0.957
	Precision	0.940	0.974	1.000	1.000	0.954	0.862
Max feature 3000	accuracy	0.956	**0.975**	0.972	0.905	0.961	0.959
	Precision	0.970	**0.975**	1.000	1.000	0.946	0.869
Scaling	accuracy	**0.967**	0.972	**0.979**	0.905	0.961	0.959
	Precision	0.964	0.943	0.946	**0.976**	0.946	0.869
Number of Characters	accuracy	0.961	0.867	0.940	**0.928**	**0.972**	**0.968**
	Precision	**0.971**	0.000	1.000	0.771	0.950	0.913

VI. Conclusion

This study demonstrated that spam detection performance depends heavily on dataset characteristics, feature engineering, and model selection, and that better feature representation and class-imbalance handling further improve accuracy and real-world reliability. The significant class imbalance between ham and spam messages posed a major challenge, often skewing results and reducing spam detection accuracy. Among the tested algorithms, SVC and MNB consistently achieved strong results, while ensemble methods like AdaBoost and Bagging showed robustness when additional features, such as number of characters features, were introduced. However, no single model outperformed across all scenarios, highlighting the importance of feature selection and data preprocessing in achieving balanced precision and accuracy.

Despite these contributions, this study has several limitations. The dataset is relatively small and exhibits a substantial class imbalance, which may affect the generalizability of the findings despite the use of class weighting. In addition, the analysis is limited to classical machine-learning models; while this choice supports efficiency and real-world deployability, it does not include transformer-based deep learning approaches that currently dominate state-of-the-art text classification. Finally, the feature set is constrained to textual and character-level attributes and does not incorporate metadata or multimodal information that could further enhance detection performance.

Future research should aim to improve model resilience against evolving spam tactics by integrating real-time detection capabilities and online learning approaches that continuously update with new spam patterns. Enhancing feature engineering — such as incorporating metadata (timestamps, sender behavior, and device information) could further strengthen classification performance. Additionally, exploring hybrid or ensemble approaches that combine the strengths of different algorithms may yield more robust and adaptive spam detection systems capable of maintaining high accuracy and precision in dynamic environments.

Statements and Declarations

Competing Interests: The authors declare no competing interests related to the content of this study. The research was conducted with full adherence to ethical guidelines and there are no financial or non-financial conflicts of interest that could have influenced the outcomes or interpretation of this work.

Funding: This research received no external funding.

References

[1] M. Crawford, T. M. Khoshgoftaar, J. D. Prusa, A. N. Richter, and H. Al Najada, “Survey of review spam detection using machine learning techniques,” J Big Data, vol. 2, no. 1, Dec. 2015, doi: 10.1186/s40537-015-0029-9.

[2] G. Alkhodhairy and K. Saleem, “Machine learning algorithm for detecting suspicious email messages using Natural Language Processing NLP,” Alexandria Engineering Journal, vol. 128, pp. 153–165, Sep. 2025, doi: 10.1016/j.aej.2025.04.067.

[3] S. Rao, A. K. Verma, and T. Bhatia, “A review on social spam detection: Challenges, open issues, and future directions,” Expert Syst Appl, vol. 186, p. 115742, 2021, doi: https://doi.org/10.1016/j.eswa.2021.115742.

[4] X. Wang, “Spam Filtering in the Modern Era: A Review of Machine Learning, Deep Learning, and System Comparisons,” INSTICC, Jul. 2025, pp. 451–458. doi: 10.5220/0013526000004619.

[5] T. Ajani and T. Ferrante, “Cyber-analytics: an examination of machine learning algorithms for spam filtering,” Issues in Information Systems, vol. 25, no. 2, pp. 203–213, 2024, doi: 10.48009/2_iis_2024_116.

[6] Y. Kontsewaya, E. Antonov, and A. Artamonov, “Evaluating the Effectiveness of Machine Learning Methods for Spam Detection,” in Procedia Computer Science, Elsevier B.V., Jul. 2021, pp. 479–486. doi: 10.1016/j.procs.2021.06.056.

[7] Z. K. Mrisho, A. Elkana Sam, and J. David Ndibwile, “Low Time Complexity Model for Email Spam Detection using Logistic Regression.” [Online]. Available: www.ijacsa.thesai.org

[8] N. Al-Shanableh Mazen, S. Alzyoud, and E. Nashnush, “Enhancing Email Spam Detection Through Ensemble Enhancing Email Spam Detection Through Ensemble Machine Learning: A Comprehensive Evaluation of Machine Learning: A Comprehensive Evaluation of Model Integration and Performance Model Integration and Performance Part of the Management Information Systems Commons.” [Online]. Available: https://scholarworks.lib.csusb.edu/ciima

[9] I. AbdulNabi and Q. Yaseen, “Spam email detection using deep learning techniques,” in Procedia Computer Science, Elsevier B.V., 2021, pp. 853–858. doi: 10.1016/j.procs.2021.03.107.

[10] A. Ghourabi and M. Alohaly, “Enhancing Spam Message Classification and Detection Using Transformer-Based Embedding and Ensemble Learning,” Sensors, vol. 23, no. 8, Apr. 2023, doi: 10.3390/s23083861.

[11] T. A. Almeida, J. M. G. Hidalgo, and A. Yamakami, “Contributions to the study of SMS spam filtering: new collection and results,” in Proceedings of the 11th ACM symposium on Document engineering, New York, NY, USA: ACM, Sep. 2011, pp. 259–262. doi: 10.1145/2034691.2034742.