WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

MessageGuard: SMS Spam Detection using ML

TAHERA ABID[1] , MUHAMMADI AZMATH[3], ANAIZA ALI[3], MARIYA FATIMA[4]*

[1] Assistant Professor,[B.Tech,M.Tech,(PhD)] Department of IT, Nawab Shah Alam Khan College of Engineering and Technology, Hyderabad, India.
[2 3 4] Department of IT, Nawab Shah Alam Khan College of Engineering and Technology, Hyderabad, India.
Email: mariyaf474@gmail.com (*Corresponding author)

ABSTRACT :

In the digital age, unsolicited and fraudulent SMS messages pose a growing threat to user privacy and security. This project focuses on developing an intelligent, web-based SMS Spam Detection system that automatically identifies and filters spam messages using machine learning techniques. The core of the system is built on the Naive Bayes algorithm, known for its efficiency and effectiveness in text classification tasks. Additionally, the platform allows flexibility by offering users the option to test multiple ML models and compare performance.

The website features a secure user authentication system, enabling users to register, log in, and interact with the application. Upon submission of an SMS message, the system processes the input and returns a prediction label either "spam" or "ham" alongside the model's confidence score. All submission data and user credentials are stored in a secure backend database, ensuring data persistence and privacy.

Designed with a clean, user-friendly interface, the platform caters to both technical and non-technical users. This project not only serves as a practical tool for spam detection but also showcases the integration of machine learning with modern web technologies to solve real-world problems. Its primary aim is to enhance the SMS experience by reducing user exposure to harmful or irrelevant content, thereby contributing to a safer and more efficient communication environment.

Keywords: SMS Spam Detection, Machine Learning, Naive Bayes, Web Application, Text Classification, User Authentication.

1.Introduction:

The increasing reliance on smartphones has led to a surge in SMS usage, but alongside this convenience comes the growing threat of spam and phishing attacks. SMS spam commonly known as *smishing* when it involves fraudulent links or attempts to steal sensitive data is a pressing cybersecurity issue that requires fast, accurate, and scalable detection mechanisms. Unlike emails, SMS messages are short, unstructured, and informal, making it harder for traditional spam filters to identify deceptive content effectively. Several research studies have addressed this issue using machine learning (ML) and natural language processing (NLP) techniques. In one such study, “SMS Spam Detection using TF-IDF and Bagging Ensemble Method”, a combination of TF-IDF-based feature extraction and Bagging classifiers was applied to build a lightweight spam classifier. The system achieved an accuracy of 96% and emphasized the importance of ensemble methods, over-sampling, and textual preprocessing steps such as tokenization and stemming. This baseline provides valuable insight into how ensemble learners and term-based feature selection contribute to detecting spam messages effectively. In another detailed comparative work, “Comparison of Machine Learning Algorithms for SMS Spam Detection**”**, multiple ML classifiers such as Naïve Bayes, SVM, and Logistic Regression were evaluated. The paper concluded that Naïve Bayes delivered the most consistent accuracy (around 97%) while also maintaining low training time, making it a practical choice for real-time applications. These findings support the core design of *Message Guard*, which employs Naïve Bayes due to its simplicity, scalability, and compatibility with probabilistic word distribution.
A broader framework was explored in “Spam Detection in SMS Using NLP and Machine Learning”, where NLP-driven feature engineering was combined with classification models. The system involved text vectorization, lemmatization, and class balancing using SMOTE. Evaluation metrics such as precision, recall, and F1-score showed promising results above 95%, reaffirming the importance of a balanced and well-preprocessed dataset. This methodology aligns closely with *Message Guard’s* modular structure, which includes preprocessing, vectorization, and a probabilistic classifier. Expanding beyond spam into smishing attacks, the work titled “Detecting Smishing Attacks Using Feature Extraction and Classification Techniques” highlights the power of NLP-driven features combined with machine learning classifiers. The study demonstrated that SVM, after careful feature extraction and selection, achieved up to 98.39% accuracy and a 99.08% F1-score. These impressive results stem from the use of benchmark datasets and the proper selection of discriminative textual features. *Message Guard* incorporates this idea by engineering features that focus not only on spammy words but also on phishing indicators such as URLs, urgency, or financial keywords. In a more cross-domain investigation, “Feature Selection-Based Spam Detection System in SMS and Email Domain**”** assessed how different feature selection techniques affect classifier performance across both SMS and email. It found that the choice of feature selection algorithm had a profound impact on training time and model accuracy. This strongly influences the design of *Message Guard*, which emphasizes selecting the most relevant features (e.g., TF-IDF weights and keyword density) to ensure fast, real-time execution in SMS-based spam filters.

Taking feature selection a step further, "An Enhanced Random Forest Approach Using CoClust Clustering" introduced a nonlinear feature clustering strategy using CoClust to improve Random Forest performance. By grouping features based on nonlinear dependencies, the model achieved enhanced accuracy and reduced CPU time. Though *Message Guard* does not directly implement Random Forest or CoClust, the emphasis on efficient dimensionality reduction and organized feature engineering plays a key role in shaping the system's streamlined performance. the paper "Support Vector Machine Algorithm for SMS Spam Classification in the Telecommunication Industry" evaluated SVMs against Naïve Bayes, KNN, and Multinomial NB using the UCI SMS Spam Collection dataset. SVM achieved 98.9% accuracy, outperforming other models in error rate and processing time. However, the study also noted the computational burden SVMs bring when applied in real-time. *Message Guard*, while recognizing SVM's robustness, instead opts for the faster and simpler Naïve Bayes to maintain real-time performance without sacrificing much accuracy. Through these diverse yet converging perspectives, it becomes evident that effective spam and smishing detection depends on a careful blend of preprocessing, smart feature selection, and choosing the right classification algorithm. Drawing on these insights, *Message Guard* is developed as a modular, scalable, and lightweight solution tailored for SMS-based spam detection using a pipeline that integrates vectorization, keyword analysis, and probabilistic classification.

2. System Analysis and Design:

2.1 Proposed Work

The proposed system is a web-based platform that detects and classifies SMS messages as either spam or legitimate (ham). The process begins with text preprocessing, where unwanted elements such as special characters, stop words, and extra spaces are removed from the message. This cleaned text is then transformed into a numerical format using a vectorization technique that captures both the importance and frequency of words, allowing the system to understand the message content in a structured way. Once the messages are preprocessed and converted, they are passed into a trained machine learning pipeline that analyzes the features and predicts whether the input text is spam or not. This predictive capability is built upon a thoroughly tested model trained on a well-labeled dataset of SMS messages. The system selects the most accurate model based on performance metrics and uses it for real-time predictions. This enables the system to make fast and reliable decisions while minimizing misclassifications.

The user-facing side of the platform is a responsive website built using modern web technologies, allowing users to interact with the system through features such as registration, login, SMS submission, and result viewing. Every user-submitted message is processed and classified instantly, and the results are stored in a secure backend database. This allows users to view a personal history of their previously analyzed messages, creating a complete and traceable record of their activity within the system. Additional components include a feedback form that enables users to submit queries or suggestions directly to the development team, and an informative "About Us" page that outlines the purpose, scope, and motivation behind the project. Overall, the proposed system is designed to be fast, scalable, secure, and easy to use, making it ideal not only for individual users but also for organizations seeking to prevent unwanted SMS spam in real-world scenarios.

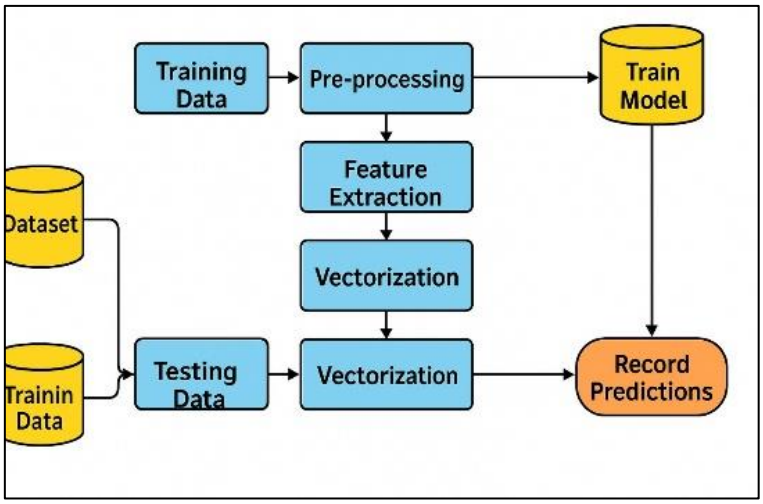
2.2 Architecture

The system architecture shown in outlines the step-by-step process involved in building and evaluating the spam detection model. It begins with the dataset, which comprises labelled SMS or email messages that indicate whether a message is spam or not. This dataset is initially divided into two main parts: training data and testing data. The purpose of this split is to train the model on one portion of the data while preserving another portion for unbiased evaluation. Both subsets undergo a pre-processing phase where the raw text is cleaned and prepared for analysis. This typically includes converting all text to lowercase, removing stopwords and special characters, performing stemming or lemmatization, and other standard text-cleaning techniques to make the data consistent and machine-readable.

Following pre-processing, the data passes through a feature extraction stage. In this phase, important textual patterns and characteristics are identified—such as word frequencies, term occurrences, or linguistic features. These features are then passed to the vectorisation module, which transforms them into numerical formats like TF-IDF vectors or bag-of-words representations. These numeric vectors serve as the input to the machine learning model.

The training data is used to train the model, enabling it to learn the underlying patterns that differentiate spam messages from legitimate ones. Once the model has been trained, it is evaluated using the testing data, which goes through the same feature extraction and vectorisation pipeline. The evaluation process helps measure the model's performance in terms of metrics like accuracy, precision, recall, and F1-score.

Finally, the trained and evaluated model is used to make predictions on new or unseen messages. These predictions are recorded, which can later be used for analysis, monitoring system behaviour, or further fine-tuning of the model. This systematic pipeline ensures that the spam detection model is developed with a strong foundation of clean, well-processed data and is evaluated effectively to ensure reliable performance in real-world scenarios.



2.3 Working of Algorithm

Naive Bayes

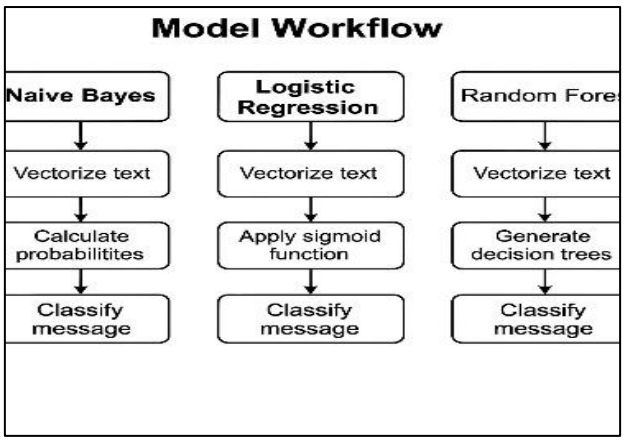
Naive Bayes is a fast and simple probabilistic classifier that performs well for text classification tasks like spam detection. It works on Bayes' Theorem, assuming that features (words) are conditionally independent given the class label. The text input is first transformed into a numerical form using techniques such as Bag-of-Words or TF-IDF. Then, the model calculates prior probabilities of each class (spam or non-spam) and the likelihood of each word given a class. Using Bayes' Theorem, the posterior probabilities for each class are computed for the input message, and the class with the highest probability is selected. Despite its strong assumption of independence, Naive Bayes performs well even with small or noisy datasets and is particularly efficient for real-time applications.

Logistic Regression

Logistic Regression is a statistical model used for binary classification problems, where the goal is to estimate the probability of a message being spam or not. The input text is preprocessed and vectorized using TF-IDF. The model assigns initial weights to each feature, computes a weighted sum of these inputs, and applies the sigmoid function to predict a probability between 0 and 1. A threshold (commonly 0.5) is then used to classify the message. The model learns by minimizing a loss function using gradient descent. Logistic Regression is valued for its interpretability, allowing insights into the importance of individual words or features in the prediction process. It works well on linearly separable data and can be regularized to prevent overfitting.

Random Forest

Random Forest is an ensemble method that builds multiple decision trees on different subsets of the training data and features. The text input is first transformed using TF-IDF. The algorithm creates several bootstrap samples from the dataset, and each sample is used to train a decision tree. Each tree independently makes a prediction, and the final output is determined by majority voting. Random Forest handles non-linear data well, is less prone to overfitting compared to individual decision trees, and is robust against noisy data. However, it is computationally intensive and less interpretable than simpler models like Logistic Regression.



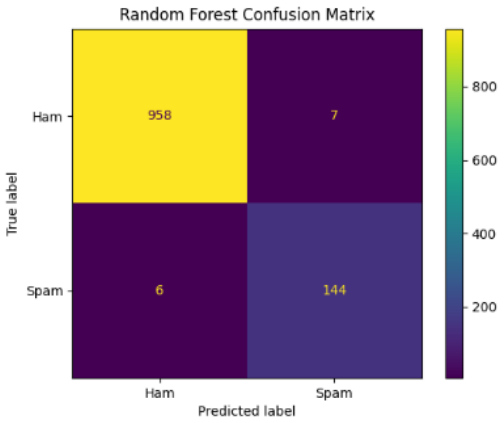
Model Selection and Evaluation

Naive Bayes was selected as the final model for this project due to its speed, simplicity, and effectiveness in handling text data. Logistic Regression and Random Forest were also tested to compare performance, and they helped validate the robustness of the chosen model. Evaluation metrics like accuracy, precision, recall, and F1-score were used to assess model performance.

4. Results:

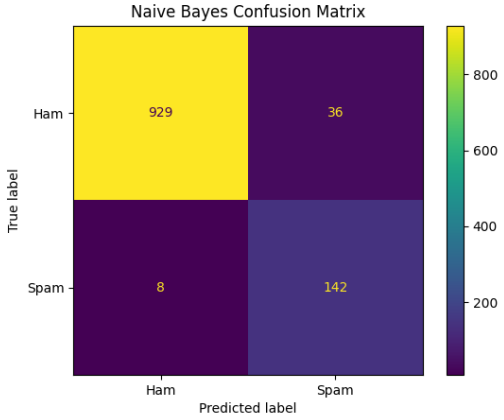
Random Forest Classifier

The Random Forest classifier performed exceptionally well on the SMS spam classification task. According to the confusion matrix, the model correctly identified 958 ham messages and 144 spam messages, while misclassifying only 7 ham messages as spam and 6 spam messages as ham. This led to an impressive accuracy of 98.31%, indicating that nearly all predictions were correct. The model also demonstrated a precision of 95.37%, meaning that the majority of messages it labeled as spam were indeed spam. Furthermore, it achieved a recall of 96.00%, suggesting it was highly effective at identifying actual spam messages. The F1-score, which balances precision and recall, was calculated at 95.68%, making Random Forest the best-performing algorithm in this evaluation. Its low false positive and false negative rates highlight its robustness and reliability for spam detection tasks.



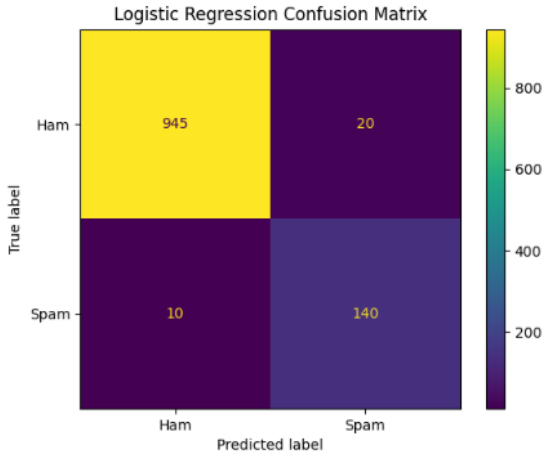
Naïve Bayes Classifier Results

The Naïve Bayes classifier showed good performance, though not as high as Random Forest. Based on the confusion matrix, it correctly predicted 929 ham messages and 142 spam messages. However, it misclassified 36 ham messages as spam and 8 spam messages as ham. The overall accuracy stood at 94.76%, which is decent but reflects the relatively higher number of false positives. The precision was 79.77%, indicating that about 20% of messages labeled as spam were actually ham — a possible drawback in real-world applications where false alarms could affect user experience. On the other hand, the classifier had a recall of 94.67%, showing that it successfully detected most of the actual spam messages. The F1-score was 86.27%, reflecting a fair trade-off between precision and recall. Naïve Bayes remains a lightweight, fast option but may not be ideal where high precision is critical.



Logistic Regression Classifier Results

Logistic Regression also delivered strong results on the dataset. As seen in the confusion matrix, it accurately predicted 945 ham messages and 140 spam messages, while misclassifying 20 ham messages as spam and 10 spam messages as ham. This resulted in an accuracy of 96.98%, positioning it as the second-best performer after Random Forest. The precision reached 87.50%, indicating that most of the predicted spam messages were correct. With a recall of 93.33%, the model was effective at identifying true spam cases. The F1-score, balancing both metrics, was 90.30%, showing that Logistic Regression maintains a good equilibrium between false positives and false negatives. While not quite as precise as Random Forest, it still offers a dependable and interpretable solution for spam classification.



5. Conclusion:

The SMS Spam Detection System developed in this project demonstrates how machine learning can be effectively applied to solve real-world communication security challenges. With the increasing use of SMS as a primary mode of communication, spam and malicious messages pose a growing threat. This project addresses the issue by leveraging supervised machine learning algorithms to accurately classify SMS messages as either spam or ham (legitimate). The system incorporates essential components such as Natural Language Processing (NLP) for text preprocessing, TF-IDF vectorization for feature extraction, and classification algorithms including Naive Bayes, Logistic Regression, and Random Forest. After thorough evaluation using metrics like precision, recall, and F1-score, the best-performing model is selected to ensure reliable and accurate predictions.

In addition to model training and evaluation, the system includes a user-friendly web interface built using Flask, with features such as secure login, a message submission interface, and a prediction history log. This makes the system practical and accessible, enabling real-time user interaction with the spam detection model. While the system has not yet been deployed to a live environment, its modular and scalable design makes it suitable for future deployment and expansion. Potential enhancements could include deep learning integration, multilingual support, mobile application development, and improved database management to support a larger user base. In conclusion, the SMS Spam Detection System successfully combines machine learning, secure web development, and user-centric design to offer an effective solution for detecting SMS spam. It achieves its core objective with high accuracy and lays a strong foundation for future development in the realm of communication security.

6. Future Enhancements & References:

6.1 Future Enhancement

The current SMS Spam Detection System has shown promising results in identifying spam effectively. However, to enhance its overall utility, future versions can include support for multiple languages, as the present model is limited to English-language datasets. Incorporating multilingual Natural Language Processing (NLP) would help cater to a more diverse population. Furthermore, transforming the system into a mobile application for Android and iOS platforms would make it more accessible, enabling real-time spam detection directly on users' devices. Real-time monitoring can help the system automatically detect spam as messages arrive, removing the need for manual input. Additionally, integrating an intelligent feedback learning mechanism would allow users to report and correct misclassified messages, facilitating continuous model refinement through periodic retraining or online learning techniques.

To improve the depth of understanding and accuracy, especially in more subtle or context-heavy messages, the system could benefit from the integration of advanced deep learning models such as LSTM or BERT. These models are known to outperform traditional machine learning algorithms when it comes to handling complex patterns in text data. In addition to this, phishing detection can be significantly enhanced by incorporating URL scanning and intent analysis to identify malicious links or attempts at social engineering. A spam reporting system can also be introduced to allow users to flag suspicious numbers, contributing to a centralized spammer database. This collective input would serve as a valuable source of updated data and improve spam blocking precision. Similarly, deploying the solution on a cloud infrastructure can ensure that it remains scalable, highly available, and efficient even under high user load, offering improved analytics and secure storage.

Broader integration opportunities such as a browser extension version of the system can help users identify spam across platforms including web-based SMS and email clients. Administrative features such as a detailed dashboard would allow monitoring of flagged numbers, user engagement, and emerging spam trends. Personalization features could also be added to give users control over spam sensitivity levels and filter preferences. To further ensure data privacy and attract privacy-conscious users, an offline or on-device processing mode could be introduced, which would ensure that sensitive SMS data never leaves the user’s device. Overall, these enhancements aim to boost the system’s accuracy, usability, adaptability, and trustworthiness, thereby making it a comprehensive and reliable solution for combating spam messages in a variety of environments.

6.2 REFERENCES

1. *https://www.inderscienceonline.com/doi/abs/10.1504/IJMLO.2022.124160*
2. *https://www.sciencedirect.com/science/article/pii/S2590005621000503*
3. *https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0272269*
4. *M. A. Abid, S. Ullah, M. A. Siddique, M. F. Mushtaq, W. Aljedaani, and F. Rustam, ''Spam SMS filtering based on text features and supervised machine learning techniques,'' Multimedia Tools Appl., vol. 81, no. 28, pp. 39853–39871, Nov. 2022.*
5. *T. Xia and X. Chen, ''A discrete hidden Markov model for SMS spam detection,'' Appl. Sci., vol. 10, no. 14, p. 5011, Jul. 2020.*
6. *Ghourabi and M. Alohaly, ''Enhancing spam message classification and detection using transformer-based embedding and ensemble learning,'' Sensors, vol. 23, no. 8, p. 3861, Apr. 2023.*
7. *N. A. M. Ariff, F. M. Husni, M. Z. Mas'ud, N. Bahaman, and E. Hamid, ''A comparison of generative and discriminative classifiers in SMS spam classification,'' in Proc. ICTEC, 2022.*
8. *Z. Ilhan Taskin, K. Yildirak, and C. H. Aladag, ''An enhanced random forest approach using CoClust clustering: MIMIC-III and SMS spam collection application,'' J. Big Data, vol. 10, no. 1, p. 38, Mar. 2023.*