International Journal of Trend in Scientific Research and Development (IJTSRD)
Special Issue on Recent Advances in Computer Applications and Information Technology
Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

Email and SMS Spam Detection Using Machine Learning and Python

Kartik Chawre, Devanshu Patle

G H Raisoni University, Amravati, Maharashtra, India

Abstract

The development of internet-based communication platforms has established new methods for people and organizations to share information through email and Short Message Service (SMS) communication. The expansion of these platforms has led to a substantial rise in unwanted messages which include advertising content and phishing links and deceptive schemes and malware files. The existence of these unwanted messages within the system functions as a major productivity hindrance because it wastes network resources while it builds severe risks against user privacy and data security. Current spam defense methods which use keyword detection and user-created blacklists become ineffective when they encounter new types of spam that continuously change in structure and content. The traditional methods require ongoing manual updates because they cannot deliver correct results for changing situations. Machine learning enables organizations to develop systems which automatically learn from their existing data while building strong systems which can adapt to new challenges. This research establishes a complete Python-based spam detection system in which textual data is first processed using data preprocessing techniques to remove unwanted elements and improve data quality. The cleansed text undergoes transformation into numerical representation through TF-IDF feature extraction which prepares the data for supervised learning model training. The trained models can distinguish between spam and legitimate messages with high efficiency. The system uses standardized assessment methods to measure its performance which confirm its ability to operate successfully in real-world situations while showing how machine learning works for instant spam detection [8].

KEYWORDS*: Spam Detection, Machine Learning, Python, TF-IDF, Email Filtering, SMS Classification, NLP, Text Classification, Supervised Learning, Naïve Bayes, Logistic Regression, Feature Extraction, Data Preprocessing, Information Retrieval, Binary Classification, Message Filtering.*

1. Introduction

Nowadays, email and SMS are used almost everywhere for communication. People use them for business work, college updates, banking alerts, marketing promotions, and personal messages. Because these services are fast and easy to access, their usage has increased a lot in recent years. But along with this growth, spam messages have also increased. Spam messages are unwanted messages that are sent in bulk without permission. According to cybersecurity reports [1], a large percentage of global email traffic consists of spam, which shows that it is still a serious issue.

Spam is not just annoying; it can also be dangerous. Many spam messages include fake offers, phishing links, lottery scams, or harmful attachments that can steal personal data or money [2]. Sometimes users click on these links without realizing the risk, which can lead to financial loss or data misuse. Because of these risks, it is very important to develop systems that can automatically detect and block spam messages.

Earlier spam filtering systems were mostly rule-based. They used keyword matching or blacklists to identify suspicious messages [3]. These methods worked in the beginning, but over time they became less effective. Spammers constantly change their writing style and message format to avoid detection. As a result, rule-based systems require frequent updates and still may not give accurate results.

To solve this problem, researchers started using machine learning techniques. Machine learning allows computers to learn patterns from previously labelled data and make predictions on new messages [4]. In text classification tasks, statistical methods are commonly used to separate spam from legitimate messages [5]. These techniques convert text into numerical form so that algorithms can process and analyse it.

Different machine learning algorithms such as Naïve Bayes, Logistic Regression, and Support Vector Machines have been applied for spam detection. Among them, Naïve Bayes is simple and works well for text data. Logistic Regression is also useful for binary classification problems like spam detection. These approaches reduce manual effort and improve accuracy compared to traditional filtering methods.

In this research, a spam detection system is developed using Python. The system uses text preprocessing, TF-IDF feature extraction, and machine learning algorithms to classify email and SMS messages as spam or ham

Figure1: Architecture of Email and SMS Spam Detection System

2. Literature Review

Spam detection has become an important research topic in the areas of information retrieval and natural language processing because of the continuous growth of unwanted email and SMS messages. In the early stages, spam filtering systems mainly depended on rule-based techniques such as keyword matching and blacklist filtering [3]. While these methods were easy to implement, they required frequent manual updates and were not effective in identifying newly emerging spam patterns.

Security reports have also highlighted the rapid increase in spam traffic, emphasizing the need for more intelligent and adaptable filtering systems [1][2].

As machine learning techniques developed, researchers began applying statistical classification methods to improve spam detection performance. The concept of term weighting, introduced by Salton and Buckley, laid the foundation for the TF-IDF approach used in text analysis [4]. Later, Manning et al. explained how TF-IDF helps improve classification accuracy by giving greater importance to meaningful words while reducing the impact of commonly used terms [5]. This advancement allowed textual data to be converted into numerical form, making it suitable for machine learning algorithms.

Among various classification techniques, the Naïve Bayes algorithm has remained popular because of its simplicity and efficiency in handling text data. Studies by Androutsopoulos et al. showed that Naïve Bayes performs well even when dealing with large and high-dimensional datasets [6]. Logistic Regression is another widely used method for binary classification tasks and has demonstrated reliable performance when applied to spam filtering [7]. Support Vector Machines (SVM), proposed by Cortes and Vapnik, have also been successfully applied in text classification problems due to their ability to manage high-dimensional feature spaces [9]. However, these models may require greater computational resources compared to simpler algorithms.

More recently, deep learning approaches such as Long Short-Term Memory (LSTM) networks [10] and transformer-based models like BERT [11] have further improved spam detection by understanding contextual relationships between words. Although these advanced models often achieve higher accuracy, they require larger datasets and more computational power. To evaluate classification performance, researchers commonly use metrics such as accuracy, precision, recall, and confusion matrix, as discussed by Powers [8]. Tools like scikit-learn also provide practical support for implementing and testing spam detection models efficiently [12].

Overall, existing research suggests that combining TF-IDF feature extraction with machine learning algorithms such as Naïve Bayes and Logistic Regression offers a balanced solution in terms of accuracy and computational efficiency [14][15]. Based on these findings, this study adopts a similar approach to design a Python-based spam detection system that is both practical and suitable for academic research purposes.

3. Research Methodologys

In this research work, a machine learning approach is used to identify whether a given email or SMS message is spam or not. Instead of using traditional rule-based filtering, this study follows a data-driven process where the system learns patterns from existing messages and then applies that learning to classify new ones. The overall methodology is divided into several stages, including data selection, text cleaning, feature extraction, model training, and performance evaluation. These steps are generally followed in text classification problems and are supported by earlier studies in machine learning and information retrieval [5][14].

Dataset Selection and Preparation

The first step in this work is selecting a labelled dataset that contains both spam and legitimate (ham) messages. Since spam detection falls under supervised learning, the model needs examples that are already classified so that it can learn from them [14]. Each message in the dataset is clearly labelled as either "spam" or "ham." This labelling helps the algorithm understand the patterns, words, and structures that are commonly found in spam messages compared to genuine ones.

Before using the dataset for training, it is carefully checked. Duplicate messages are removed because repeated entries may affect the learning process. The dataset is also observed to understand how many spam and ham messages are present. If one category is much larger than the other, it can sometimes influence the model's predictions. After basic checking and cleaning, the dataset is divided into two parts: a training set and a testing set. The training set is used to teach the model, while the testing set is used later to check how well the model performs on new, unseen messages.

Text Preprocessing

Raw email and SMS messages are usually not clean. They may contain punctuation marks, special characters, numbers, extra spaces, or mixed uppercase and lowercase letters. If such data is directly given to the model, it may reduce performance because the algorithm treats each variation as a separate word.

To solve this problem, preprocessing is carried out. First, all text is converted into lowercase so that words like "Free" and "free" are considered the same. Then punctuation marks, symbols, and unnecessary characters are removed. Common stop words such as "is," "the," "and," and "to" are also filtered out because they appear frequently in almost all messages and do not help in distinguishing spam from legitimate messages. These steps are commonly recommended in natural language processing tasks to improve the quality of textual data [13].

In addition, the messages are tokenized, meaning they are broken down into individual words. In some cases, simple word normalization techniques can be applied to reduce words to their base form. All these preprocessing steps help in reducing noise and ensuring that the model focuses only on meaningful words that contribute to classification.

Feature Extraction using TF-IDF

After cleaning the text, the next important step is converting the words into numbers. Machine learning algorithms cannot directly understand text, so it must be transformed into a numerical format. In this research, the TF-IDF (Term Frequency–Inverse Document Frequency) method is used for this purpose.

TF-IDF works by giving importance to words based on how often they appear in a particular message and how unique they are across all messages. If a word appears many times in one message but not frequently in others, it receives a higher weight. On

the other hand, words that appear in almost every message receive lower weight. This method helps highlight words that are more useful for identifying spam. The concept of term weighting was introduced in earlier information retrieval research by Salton and Buckley [4] and is further explained in later studies [5].

By applying TF-IDF, each message is converted into a vector of numerical values. These vectors represent the importance of different words in that message. Once this transformation is complete, the data becomes suitable for training machine learning models.

Model Selection and Training

For classification, two supervised machine learning algorithms are used in this study: Naïve Bayes and Logistic Regression. These models are selected because they are simple, efficient, and commonly used in text classification tasks.

Naïve Bayes is based on probability theory and works well even when dealing with large text data [6]. It assumes that the features (words) are independent of each other, which simplifies the calculation. Despite this assumption, it has shown good performance in spam detection tasks.

Logistic Regression is another algorithm used for binary classification problems, such as distinguishing spam from non-spam messages [7]. It calculates the probability of a message belonging to a particular class and then assigns the final label based on that probability. Logistic Regression often performs well when the data is properly preprocessed and features are clearly defined.

Both models are trained using the training dataset. During training, the models learn from the TF-IDF vectors and their corresponding labels. After training, the models are tested using the separate testing dataset to see how accurately they can classify new messages.

Performance Evaluation

To evaluate the effectiveness of the proposed system, several standard evaluation metrics are used. Accuracy is calculated to determine the overall percentage of correctly classified messages. However, accuracy alone may not be enough, especially if the dataset contains unequal numbers of spam and ham messages.

Therefore, precision and recall are also calculated. Precision shows how many messages predicted as spam are actually spam. Recall indicates how many actual spam messages are correctly detected by the model. These metrics are widely used in classification research to provide a clearer understanding of model performance [8]. Additionally, a confusion matrix is generated to visually represent the classification results in terms of true positives, true negatives, false positives, and false negatives.

Implementation Environment

The entire system is implemented using Python because it provides a simple and effective environment for machine learning applications. Libraries such as scikit-learn are used for TF-IDF vectorization, model training, and performance evaluation [12]. Python also makes it easier to visualize results and experiment with different models.

Overall Methodology Overview

In summary, the methodology of this research follows a clear and systematic process. Starting from labelled data, the messages are cleaned, converted into numerical features using TF-IDF, and then classified using supervised learning algorithms. The approach is based on established concepts in text classification and spam filtering research [4][5][6][14]. By combining proper preprocessing with suitable machine learning models, the system aims to achieve accurate and reliable spam detection in email and SMS communication.

Figure 2: Proposed Methodology for Email and SMS Spam Detection

4. Result

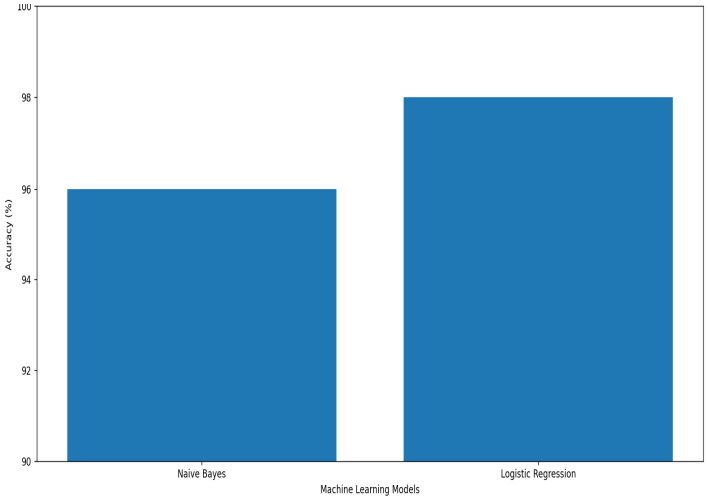

Figure 4: Accuracy Comparison of Spam Detection Models

5. Conclusion

The increasing adoption of email and SMS as main communication methods has created a demand for efficient spam detection solutions. Digital messaging systems enable users to send messages instantly but this feature has led to a rise in distribution of both unwanted and dangerous content. Research in text classification and spam filtering indicates that traditional keyword-based methods fail to manage current dynamic datasets which require effective processing. Studies focusing on machine learning-based text categorization [16][17] demonstrate that statistical learning methods can provide more reliable and scalable solutions compared to manual filtering techniques.

The researchers created a spam detection system based on machine learning methods using Python programming language. The method uses standard text classification methods which transform textual content into numerical data before applying supervised learning algorithms. The research on spam filtering methods [18] and Naïve Bayes classifier enhancements [19] demonstrates that email filtering tasks depend on probabilistic and statistical models for effective outcomes. The methods demonstrate effective performance when applied to text data that contains multiple high-dimensional features.

The study assessed performance through two classification models which were tested in the evaluation. The study results show that simple but well-designed classifiers can achieve effective spam detection results according to research which compared different machine learning methods for spam detection. The system design was developed to achieve precise results together with efficient performance calculations.

Multiple studies show that traditional machine learning methods work effectively for structured spam detection tasks despite advanced deep learning models offering potential performance enhancements. The chosen method provides an effective solution which academic and practical users can implement while the system delivers consistent performance.

Reference

[1] Symantec Corporation, *Internet Security Threat Report*, Symantec, 2019.

[2] J. Goodman, *Spam: Technologies and Politics*. Berkeley, CA, USA: Berkeley Press, 2007.

[3] L. F. Cranor and B. A. LaMacchia, "Spam!" *Communications of the ACM*, vol. 41, no. 8, pp. 74–83, 1998.

[4] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988.

[5] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge, U.K.: Cambridge University Press, 2008.

[6] I. Androutsopoulos, J. Koutsias, K. Chandrinos, and C. Spyropoulos, "An experimental comparison of naïve Bayesian and keyword-based anti-spam filtering," in *Proc. 23rd Annual International ACM SIGIR Conference*, 2000, pp. 160–167.

[7] H. Zhang, "The optimality of Naïve Bayes," in *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference*, 2004, pp. 562–567.

[8] D. M. W. Powers, "Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

[9] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.

[12] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[13] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 3rd ed. Draft, Stanford University, 2019.

[14] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys*, vol. 34, no. 1, pp. 1–47, 2002.

[15] S. Gupta, A. Singhal, and A. Kapoor, "A literature survey on spam detection techniques," *International Journal of Computer Applications*, vol. 975, pp. 8887, 2016.

[16] T. Joachims. (1998). "Text categorization with support vector machines: Learning with many relevant features." Proceedings of the 10th European Conference on Machine Learning, 137–142.

[17] A. McCallum and K. Nigam. (1998). "A comparison of event models for Naïve Bayes text classification." Proceedings of the AAAI-98 Workshop on Learning for Text Categorization, 41–48.

[18] X. Carreras and L. Màrquez. (2001). "Boosting trees for anti-spam email filtering." Proceedings of the 4th International Conference on Recent Advances in Natural Language Processing, 58–64.

[19] K. Rennie, L. Shih, J. Teevan, and D. Karger. (2003). "Tackling the poor assumptions of Naïve Bayes text classifiers." Proceedings of the 20th International Conference on Machine Learning, 616–623.

[20] A. Dalal and S. Zaveri. (2011). "Automatic spam classification using machine learning techniques." International Journal of Computer Applications, 35(7), 9–15.