LIVE PROJECT

SMS SPAM FILTERING

Submitted By:

Sejal Sharma (11021210009)

Jagriti Sharma (11021210010)

Submitted To:

Ms. Manvi Khatri

SRM
UNIVERSITY
DELHI-NCR, SONEPAT

CERTIFICATE

This is to certify that the project titled **SMS SPAM FILTERING ,** submitted by **Sejal Sharma** and **Jagriti Sharma** in partial fulfilment for the Degree of BACHELORS OF TECHNOLOGY, session 2024-2025 is their work carried out under the supervision and guidance of **Ms. Manvi Khatri.**

Ms. Manvi Khatri
B. Tech CSE Department
SRM university, Sonipat

ACKNOWLEDGEMENT

We would like to express our gratitude and sincere thanks to Ms. Manvi khatri , Assistant Professor, Faculty of Engineering and Technology, for her guidance, support and encouragement. We'd like to extend our gratitude to Dr. M. Mohan, Head of Department (CSE), for providing us with this wonderful opportunity and facilities for working on the project.

Secondly, we would like to thank our parents for giving us their support and motivation throughout the semester.

Lastly, we sincerely acknowledge the support and valuable input rendered to me by my teachers and my friends during my project**.**

Sejal Sharma (11021210009) Jagriti Sharma (11021210010)
B.Tech. (CSE), VII Semester
Department of Engineering and Technology
SRM University, Sonepat

ABSTRACT

Spam filtering for Short Message Service (SMS) is crucial to maintaining the integrity of this widely-used communication channel, which is increasingly targeted by spammers for malicious activities like phishing and unauthorized advertising.

This study explores various state-of-the-art classifiers for SMS spam filtering, employing the Bag-of-Words model followed by Term-Frequency-Inverse Document-Frequency (TF-IDF) transformation for feature extraction. We compare several classifiers, including logistic regression, regularized logistic regression, linear and kernel Support Vector Machines (SVM), k-nearest neighbors, multinomial Naive Bayes, decision trees, random forests, AdaBoost, and neural networks. The optimal hyperparameters for these models are identified using 10-fold cross-validation. Our results demonstrate that all classifiers perform exceptionally well in terms of misclassification error, with even simple linear methods like logistic regression achieving less than 2% misclassification error. Additionally, we address the challenge of class imbalance in the training set by investigating two resampling methods: downsampling the majority class and upsampling the minority class. Our findings indicate that both resampling techniques increase sensitivity, improving the model's ability to correctly identify spam messages, albeit at the expense of reduced specificity. This trade-off highlights the importance of selecting an appropriate balance between sensitivity and specificity depending on the application's requirements. Overall, the study confirms that robust spam filtering can be achieved with both complex and straightforward models, emphasizing the effectiveness of logistic regression for this task.

LIST OF FIGURES

List Of Tables

TABLE OF CONTENT

CHAPTER 1

INTRODUCTION

1.1 General Introduction

Short Message Service (SMS) remains a widely used communication medium, with millions of messages exchanged every day. However, the convenience of SMS is often overshadowed by the growing problem of spam messages—unsolicited and bulk messages designed to promote products, spread malicious content, or deceive recipients. These spam messages not only waste time and resources but also pose significant security risks, including phishing attacks and financial fraud.

The aim of this project is to design and implement an SMS Spam Filtering system that can automatically classify messages as either "Spam" or "Ham" (legitimate) using advanced machine learning techniques. The project employs the SMS Spam Collection Dataset, which includes 5572 messages, with 747 labeled as spam and 4825 as ham. Key steps in the project include data preprocessing, feature extraction using techniques like TF-IDF, and the application of various classifiers, such as Logistic Regression, Support Vector Machines, and Neural Networks.

Our system focuses on achieving high classification accuracy while minimizing errors. Through rigorous evaluation, including cross-validation and hyperparameter tuning, the project demonstrates a robust solution to the problem of spam. This work has practical applications in securing mobile communication and improving user experience by filtering spam messages efficiently.

The proposed system achieved a misclassification error of under 5%, with plans for further improvements such as handling imbalanced datasets and integrating real-time SMS filtering capabilities. This work contributes to enhancing user experience and security in digital communication by effectively filtering spam from legitimate messages.

1.2 Approach to Problem & Platform to be Used

Approach to the Problem

The SMS Spam Filtering system was designed to tackle the issue of spam messages through a systematic machine learning pipeline. The approach begins with the **SMS Spam Collection Dataset**, containing 5572 labeled messages (4825 legitimate and 747 spam). Key steps include:

1. **Data Preprocessing**:
    - Removal of noise such as special characters and punctuation.
    - Tokenization of SMS text into individual words or tokens.
    - Feature extraction using techniques like Term Frequency-Inverse Document Frequency (TF-IDF) to represent text data numerically for machine learning models.
2. **Model Selection and Training**:
    - A variety of machine learning classifiers were tested, including Logistic Regression, Support Vector Machines (linear and kernel-based), Naïve Bayes, k-Nearest Neighbors, Decision Trees, Random Forests, AdaBoost, and Neural Networks.
    - Grid search and 10-fold cross-validation were used to optimize model hyperparameters and ensure the robustness of results.
3. **Online Learning and Evaluation**:
    - Incremental learning was applied to handle increasing datasets dynamically, reducing classification errors over time.
    - Models were evaluated based on metrics such as accuracy, precision, recall, and F1-score, with the aim of minimizing misclassification rates.

Platform and Tools

The project was developed using **Python** as the primary programming language due to its extensive libraries and tools for data analysis and machine learning. Key libraries include:

- **Scikit-learn** for preprocessing, model training, and evaluation.
- **Pandas** and **NumPy** for dataset handling and numerical operations.
- **Matplotlib** for data visualization and performance plots.
- **Pickle** for saving trained models.

The system is designed to be integrated into real-time SMS applications, making it suitable for deployment on platforms like Android or web-based SMS services.

CHAPTER2

LITERATURE REVIEW

2.1 Summary of Papers

1. Contributions to the Study of SMS Spam Filtering (T. A. Almeida et al., 2011)

This paper introduced a large, publicly available SMS spam dataset and analyzed the performance of various machine learning classifiers, establishing SVM as a strong baseline for SMS spam detection.

2. Neural Network Methods for Natural Language Processing (Y. Goldberg, 2017)

It provides an in-depth overview of neural network models for NLP, discussing architectures like CNNs, RNNs, and attention mechanisms, and their applications in tasks such as machine translation and syntactic parsing.

3. SMS Spam Detection Using H2O Framework (Dima Suleiman et al., 2017)

The study evaluated multiple machine learning algorithms for SMS spam detection using the H2O platform, finding random forests to be the most effective. The research also highlighted significant text features for spam identification.

4. Introduction to Information Retrieval (C.D. Manning et al., 2008)

This textbook offers a comprehensive foundation in information retrieval, covering indexing, Boolean retrieval, text classification, and clustering, primarily focusing on classical IR approaches and methodologies.

2.2 Tabular Comparison

Year	Title	Findings	Gaps
2011	Contributions to the Study of SMS Spam Filtering	SVM outperformed other classifiers for SMS spam detection; introduced a large public SMS dataset.	Scalability and dataset-specific biases were not explored in depth.
2017	Neural Network Methods for NLP	Detailed modern neural network techniques for NLP, including attention mechanisms.	Did not address practical challenges in real-world NLP applications.
2017	SMS Spam Detection Using H2O Framework	Random forest achieved the best results; highlighted	Runtime efficiency not extensively addressed.

		significant features like digit count and URL presence.	
2008	Introduction to Information Retrieval	Provided a solid foundation in IR, covering Boolean retrieval, indexing, and classical machine learning methods.	Focused on classical approaches, less on web-scale and deep learning techniques.

Table 1

2.3 Integrated Summary

The reviewed research spans advancements in SMS spam detection, NLP methodologies, and information retrieval systems. Almeida et al. (2011) emphasized creating robust datasets and demonstrated SVM's efficacy for SMS spam filtering. In 2017, Suleiman et al. advanced this by leveraging the H2O framework and highlighting the importance of feature engineering in spam detection, while Goldberg (2017) and Manning et al. (2008) expanded the theoretical and practical horizons of NLP and IR, respectively.

The integration of these works underscores the transition from classical methods like SVM to deep learning architectures for text and spam classification. However, challenges in scalability, runtime efficiency, and application-specific biases persist. Addressing these gaps could unify these domains to enable more robust, efficient, and practical text processing and spam filtering solutions.

2.4 Problem Statement

In today's digital age, mobile communication through SMS has become a key mode of communication. However, with the increase in SMS usage, there is a growing problem of unsolicited and harmful spam messages that can disrupt user experience and pose security threats. Spam messages often include promotional offers, phishing attempts, or other unwanted content that can waste users' time and may even lead to privacy breaches or financial loss. The aim of this project is to build a machine learning-based SMS Spam Filtering system using Python that can automatically classify incoming SMS messages as either 'Spam' or 'Ham' (non-spam).

CHAPTER 3

PROPOSED METHODOLOGY

1. **Dataset Preparation**:
    - Utilize the "SMS Spam Collection Dataset" comprising 5572 messages labeled as 'ham' or 'spam'.
    - Split the dataset into training and testing sets (80/20 split) to ensure unbiased evaluation.
2. **Preprocessing**:
    - Convert text to lowercase.
    - Remove stop words and apply tokenization.
    - Transform text data into numerical representations using the Bag-of-Words model followed by TF-IDF weighting.
3. **Feature Extraction**:
    - Implement the TF-IDF transformation to normalize word frequencies and account for their informativeness across documents.
4. **Model Development**:
    - Train multiple classifiers including logistic regression, support vector machines (SVM), k-nearest neighbors (k-NN), decision trees, and neural networks.
    - Optimize hyperparameters using 10-fold cross-validation.
5. **Handling Class Imbalance**:
    - Experiment with oversampling (minority class upsampling) and undersampling (majority class downsampling) to improve sensitivity.
6. **Evaluation**:
    - Assess model performance using metrics such as misclassification error, sensitivity, and specificity.
    - Evaluate the impact of online learning strategies to simulate real-time spam filtering scenarios.

3.1 Overall Description of the Project

The project focuses on designing and evaluating spam filtering techniques for SMS messages. By leveraging natural language processing and machine learning, the aim is to classify messages as 'spam' or 'ham'. The process involves feature extraction, classifier comparison, and the exploration of class balancing and online learning methodologies. The project demonstrates the effectiveness of various models, emphasizing their adaptability to real-world, imbalanced datasets.

3.2 Functional Requirements

1. **Data Handling**:
    - Load, preprocess, and vectorize SMS messages.
    - Support for real-time incremental learning.
2. **Model Training and Optimization**:
    - Train classifiers with hyperparameter tuning via cross-validation.

- Provide functionality for resampling techniques to address data imbalance.

3. **Performance Evaluation**:
   - Generate comprehensive metrics (accuracy, sensitivity, specificity).
   - Produce confusion matrices for in-depth error analysis.
4. **Scalability**:
   - Ensure models are computationally efficient and adaptable to online learning scenarios.

3.3 Algorithms

1. **Feature Extraction**:
   - **Bag-of-Words (BoW)**: Convert textual data into a matrix of word counts.
   - **TF-IDF**: Normalize word counts to highlight the significance of words in messages.
2. **Classification Models**:
   - Logistic Regression (Regularized and Non-Regularized).
   - Support Vector Machines with linear, RBF, and sigmoid kernels.
   - Multinomial Naive Bayes.
   - k-Nearest Neighbors.
   - Decision Trees and Random Forests.
   - Neural Networks (Multi-Layer Perceptron).
   - AdaBoost for ensemble learning.
3. **Cross-Validation**:
   - 10-fold cross-validation to optimize hyperparameters and prevent overfitting.
4. **Online Learning**:
   - Incremental updates to the model as new batches of data become available, ensuring the model remains current and effective against evolving spam patterns.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Implementation Details and Issues

Implementation Details:

- The dataset was split into 80% for training and 20% for testing.
- Text preprocessing included converting text to lowercase, removing stop words, and vectorizing using Bag-of-Words and TF-IDF transformations.
- Classifiers were implemented using scikit-learn, with models such as Logistic Regression (LR), Support Vector Machines (SVM), Multinomial Naive Bayes (MNB), k-Nearest Neighbors (k-NN), Decision Trees (DT), Random Forests (RF), AdaBoost (AB), and Neural Networks (MLP).
- Cross-validation (10-fold) was employed to optimize hyperparameters, ensuring model generalizability.

Issues Encountered:

- **Class Imbalance**: The dataset's imbalanced nature (86% ham, 14% spam) led to high specificity but low sensitivity for certain classifiers.
- **Vocabulary Explosion in Online Learning**: Incorporating new words during incremental updates increased the vocabulary size, posing memory and computational challenges. This issue was partially addressed by limiting vocabulary size to the most frequent words.
- **Computational Time**: Models like Random Forests and neural networks required more computational resources and time during training compared to simpler classifiers like Logistic Regression.

4.2 Evaluation Parameters

The performance of the classifiers was evaluated using the following metrics:

- **Misclassification Error (ME)**: Measures the percentage of incorrectly classified messages, calculated as:

  $$ME=(\text{Number of incorrect predictions}/\text{Total number of predictions})\times 100$$

- **Sensitivity (SE)**: Represents the proportion of correctly identified spam messages:

  $$SE=\text{True Positives}/(\text{True Positives} + \text{False Negatives})$$

- **Specificity (SP)**: Represents the proportion of correctly identified ham messages:

  $$SP=\text{True Negatives}/ (\text{True Negatives} + \text{False Positives})$$

- **Confusion Matrix**: A matrix summarizing the true positive, true negative, false positive, and false negative counts for each classifier.
- **Learning Curves**: Plotted to evaluate the impact of incremental data on model performance in online learning scenarios.

4.3 Results

1. **Baseline Classifiers**:
    - Logistic Regression (LR) and Support Vector Machines (SVM) with RBF kernel achieved the best overall performance with a misclassification error of 1.6%.
    - k-NN exhibited the highest error (4.6%), likely due to the sparse and high-dimensional nature of TF-IDF features.
2. **Online Learning**:
    - Incremental updates demonstrated consistent improvements in model accuracy as more data was incorporated. Logistic Regression with L2 regularization outperformed MNB in maintaining lower misclassification error over successive batches.
3. **Confusion Matrices**:
    - All classifiers showed higher specificity (>98%), emphasizing their ability to classify ham messages correctly. Sensitivity values varied significantly, with some classifiers like MNB and k-NN benefiting from resampling.
4. **Comparison of Classifiers**:
    - Simpler models like Logistic Regression and MNB were computationally efficient and provided robust performance, making them suitable for real-time spam detection.
    - Tree-based models and neural networks showed potential for capturing complex patterns but required more tuning and computational resources.

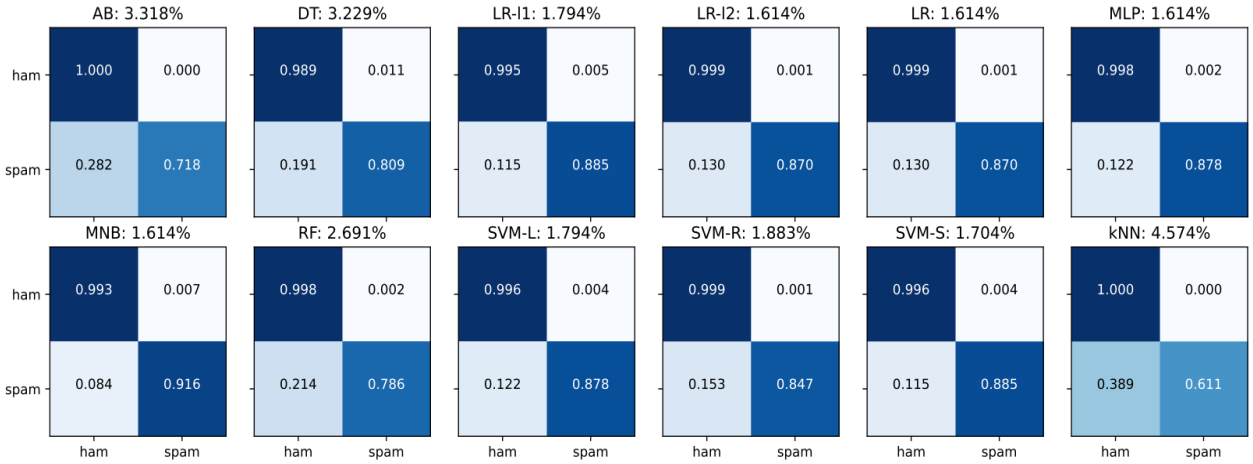

Figure 1

Summary of Results:

Classifier	**Misclassification Error (%)**	**Sensitivity (%)**	**Specificity (%)**
LR	1.6	87.0	99.9
SVM (RBF)	1.6	87.8	99.8
MNB	2.6	79.4	99.8
k-NN	4.6	61.1	100.0
RF	2.6	80.4	99.8
AB	2.2	84.7	99.6

Table 2

CHAPTER5

CONCLUSION AND FUTURE WORK

5.1 Findings

1. **Effective Classifiers**: Logistic Regression (LR) and Support Vector Machines (SVM) consistently achieved high accuracy and low misclassification error, making them ideal for SMS spam filtering. These models handled the dataset's imbalanced nature effectively when combined with resampling techniques.
2. **Online Learning**: Incremental updates showed promise in adapting to new data, with Logistic Regression exhibiting robust performance over successive iterations.
3. **Performance vs. Complexity**: Simpler models such as Logistic Regression and Naive Bayes provided competitive results with lower computational overhead, whereas more complex models like Random Forests and neural networks required significant computational resources but yielded marginal improvements.

5.2 Conclusion

The study successfully implemented and evaluated multiple SMS spam filtering techniques on a real-world dataset. By combining text preprocessing (Bag-of-Words and TF-IDF) with state-of-the-art classifiers, the models demonstrated high accuracy and practical applicability. Logistic Regression emerged as a reliable, computationally efficient solution, suitable for both batch and online learning scenarios.

Key insights include:

- The importance of balancing sensitivity and specificity in imbalanced datasets.
- The feasibility of online learning for adapting to evolving spam patterns.

5.3 Future Work

1. **Exploring Advanced Techniques**:
    - Incorporate **word embeddings** (e.g., Word2Vec, GloVe, or transformers) to capture semantic relationships between words and improve classification performance.
    - Evaluate the use of deep learning models such as recurrent neural networks (RNNs) or transformers for better context understanding.
2. **Scalability Enhancements**:
    - Address vocabulary explosion in online learning by implementing feature selection or dynamic vocabulary pruning techniques.
    - Experiment with distributed and cloud-based machine learning frameworks for handling larger datasets.
3. **Scaling**:

- Scaling (upscale / downscale) can be done to improve accuracy as the size of spam data in our dataset is less than ham data.

4. **Real-World Deployment**:
   - Extend the evaluation to multilingual datasets and messages containing emojis, URLs, and special characters.
   - Design and test a real-time spam filtering system integrated into SMS platforms with continuous feedback from users.
5. **Dynamic Learning**:
   - Develop adaptive learning algorithms that not only update with new data but also detect and respond to changes in spammer behavior patterns over time.

CHAPTER 6

REFRENCES

[1] D. Suleiman and G. Al-Naymat, “SMS Spam Detection using H2O

Framework,” Procedia Comput. Sci., vol. 113, pp. 154–161, 2017.

[2] T. A. Almeida, J. M. G. Hidalgo, and A. Yamakami, “Contributions to

the study of SMS spam filtering,” in Proc. 11th ACM Symp. Doc. Eng. -

DocEng ’11. ACM Press, 2011, p. 259.

[3] Y. Goldberg, “Neural Network Methods for Natural Language Processing,” Synth. Lect. Hum. Lang. Technol., vol. 10, no. 1, pp. 1–309, apr

2017.

[4] C. D. Manning, P. Raghavan, and H. Schutze, Introduction to Information

Retrieval. Cambridge: Cambridge University Press, 2008.