SMS Spam Detection

A Project Report

submitted in partial fulfillment of the requirements

of

AICTE Internship on AI: Transformative Learning
with
TechSaksham – A joint CSR initiative of Microsoft & SAP

by

Chaitanya Ashish Kelkar, chaitanyaakelkar57@gmail.com

Under the Guidance of

Jay Rathod

ACKNOWLEDGEMENT

We would like to express our heartfelt gratitude to everyone who supported us, directly or indirectly, throughout the course of this work.

First and foremost, we extend our deepest thanks to our supervisor, Jay Rathod, for his invaluable mentorship and guidance. His insightful advice, constructive feedback, and steadfast encouragement have been a constant source of inspiration and innovation, significantly contributing to the successful completion of this project. His confidence in our abilities has been a driving force throughout this journey. It has been a true privilege to work under his guidance during this program, as he provided not only project-specific support but also valuable advice across various aspects of the program. His lessons have not only enhanced the project's outcomes but have also shaped me into a more responsible and professional individual.

ABSTRACT

SMS spam has become a major concern in mobile communication, overwhelming users with unsolicited and potentially harmful messages. The aim of this project is to develop a robust system for classifying SMS messages as either **spam** or **ham** (legitimate) using the **Naive Bayes** algorithm. The main objectives are to reduce the prevalence of spam messages, improve user experience, and enhance mobile security.

The methodology includes several stages: first, the raw SMS data is preprocessed through tokenization, stopword removal, and stemming/lemmatization. The cleaned text is then transformed into numerical feature vectors using techniques like **TF-IDF** or **Bag of Words**. The Naive Bayes classifier, a probabilistic model based on Bayes' Theorem, is used for training on a labeled dataset of SMS messages. The classifier predicts whether a message is spam or ham based on the extracted features.

Key results show that the Naive Bayes model performs well in distinguishing between spam and ham messages, achieving high accuracy, precision, and recall, making it an effective solution for SMS spam detection. The system provides real-time classification of incoming messages and can be integrated into mobile devices for better security.

In conclusion, this project demonstrates the applicability of the Naive Bayes algorithm in the domain of spam detection. While the system performs effectively, future work can focus on real-time deployment, multilingual support, and dynamic learning to improve adaptability to emerging spam trends. This solution contributes to enhancing the user experience by efficiently filtering out unwanted and malicious SMS messages, providing better privacy and security in mobile communication.

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1

Introduction

1.1Problem Statement:

The increasing prevalence of spam messages in mobile communication poses significant challenges, including disrupting user experience, compromising security through phishing and fraudulent schemes, and burdening organizations with additional costs. Traditional filtering methods are insufficient to address the evolving nature and growing volume of spam, necessitating an intelligent and scalable solution for effective detection and prevention., reducing the risk of spam messages and enhancing communication efficiency.

1.2Motivation:

This project was chosen to address the growing challenge of spam messages, which disrupt user experience and pose security risks such as phishing and fraud. With the rapid increase in mobile communication, there is a need for effective, adaptable solutions for spam detection.

Potential Applications and Impact

- **Personal Platforms**: Enhances user experience by filtering spam in messaging apps.
- **Enterprise Solutions**: Protects businesses from phishing and fraudulent messages.
- **Telecom Providers**: Prevents spam at the network level.
- **Global Use**: Expands to detect spam in multiple languages.

This project aims to improve communication security, protect users, and reduce the economic impact of spam through scalable and efficient detection methods.

1.3Objective:

- **Develop an Effective Spam Detection System**: Build a model capable of classifying SMS messages as spam or legitimate using machine learning and natural language processing techniques.
- **Data Preprocessing and Feature Extraction**: Implement methods for cleaning and preparing the SMS dataset, including tokenization, text normalization, and feature extraction using techniques like TF-IDF.
- **Evaluate Classification Algorithms**: Test and compare various classification algorithms, such as Naïve Bayes, Support Vector Machines

(SVM), and Random Forest, to determine the most accurate and efficient model.

- **Performance Evaluation**: Assess the model's performance based on metrics like accuracy, precision, recall, and F1 score to ensure reliable spam detection.

Build a Scalable Solution: Ensure the developed system can handle large datasets and be integrated into real-world applications, such as mobile messaging platforms or enterprise systems.

1.4Scope of the Project:

Scope:

- **SMS Spam Detection**: The project focuses on detecting spam messages specifically within SMS communication, using text classification techniques based on machine learning and natural language processing.
- **Machine Learning and NLP Techniques**: The project utilizes algorithms like Naïve Bayes, Support Vector Machines (SVM), and Random Forest, along with NLP methods such as tokenization and TF-IDF for feature extraction.
- **Dataset**: The project will use publicly available SMS datasets for training and evaluation, aiming to achieve high accuracy in distinguishing spam from legitimate messages.
- **Evaluation Metrics**: The performance of the developed model will be assessed using common evaluation metrics, including accuracy, precision, recall, and F1 score.

Limitations:

- **Dataset Constraints**: The quality and size of the dataset may affect the performance and generalizability of the model, as real-world datasets may vary in content and language.
- **Language Limitation**: The system will primarily be focused on English-language SMS messages, limiting its applicability to multilingual environments.
- **Static Models**: The models developed will be trained on historical data and may not adapt to new, unseen spam tactics or emerging types of messages without retraining.
- **Scope of Detection**: The system will detect only text-based spam messages and will not address multimedia or non-textual spam (e.g., images, videos).

CHAPTER 2

Literature Survey

2.1 Review of relevant literature.

The detection of SMS spam has been a widely researched domain, given its significance in preventing fraud and maintaining privacy. Early approaches primarily focused on rule-based systems, where keyword matching and blacklists formed the basis of spam identification. As the volume and complexity of spam messages increased, machine learning (ML) techniques became a preferred choice. Recent studies emphasize the integration of natural language processing (NLP) for tokenizing text and extracting features that capture the semantic and syntactic properties of messages. Research by Almeida et al. (2013) and others has demonstrated the efficacy of supervised learning techniques like Naive Bayes, Support Vector Machines (SVM), and decision trees in classifying SMS as spam or legitimate (ham).

2.2 Existing models, techniques, methodologies.

Traditional Machine Learning Models:

- Naive Bayes: Commonly used for text classification due to its simplicity and efficiency with large datasets.
- SVM: Effective in separating spam and ham messages using hyperplanes in high-dimensional space.
- Random Forest: Provides robust results by combining multiple decision trees for classification tasks.

NLP Techniques:

- Tokenization: Splitting text into words or phrases for processing.
- Stemming and Lemmatization: Reducing words to their base or root forms for feature extraction.
- TF-IDF (Term Frequency-Inverse Document Frequency): Weighing words based on their importance within the message corpus.
- Deep Learning Models:
- Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM): Effective for capturing sequential dependencies in text data.
- Transformers (e.g., BERT): Provide state-of-the-art performance in text classification by leveraging attention mechanisms.

Hybrid Approaches:

- Combining NLP-based feature extraction with traditional ML classifiers to improve accuracy.

2.3 Highlight the gaps or limitations in existing solutions and how your project will address them.

Challenges in Handling Short Text:
SMS messages are typically short and lack sufficient context, making feature extraction challenging. Many existing models fail to fully capture the nuances of short text.
Proposed Solution: Employ advanced NLP techniques tailored for short text, such as n-gram modeling and word embeddings like Word2Vec or GloVe, to better understand the context.

Imbalanced Datasets:
Real-world SMS datasets often have a disproportionate number of ham messages compared to spam, which affects model performance.
Proposed Solution: Address data imbalance using techniques like SMOTE (Synthetic Minority Oversampling Technique) or class-weighted loss functions during training.

Dynamic Nature of Spam:
Spam messages evolve over time with new patterns and tactics, rendering static models less effective.
Proposed Solution: Incorporate adaptive learning techniques or semi-supervised learning to continually update the model based on new data.

High False Positive Rates:
Some existing models have high false positive rates, leading to legitimate messages being flagged as spam.
Proposed Solution: Optimize hyperparameters and use ensemble methods to reduce misclassification rates.

CHAPTER 3

Proposed Methodology

3.1 System Design

Provide the diagram of your Proposed Solution and explain the diagram in detail.

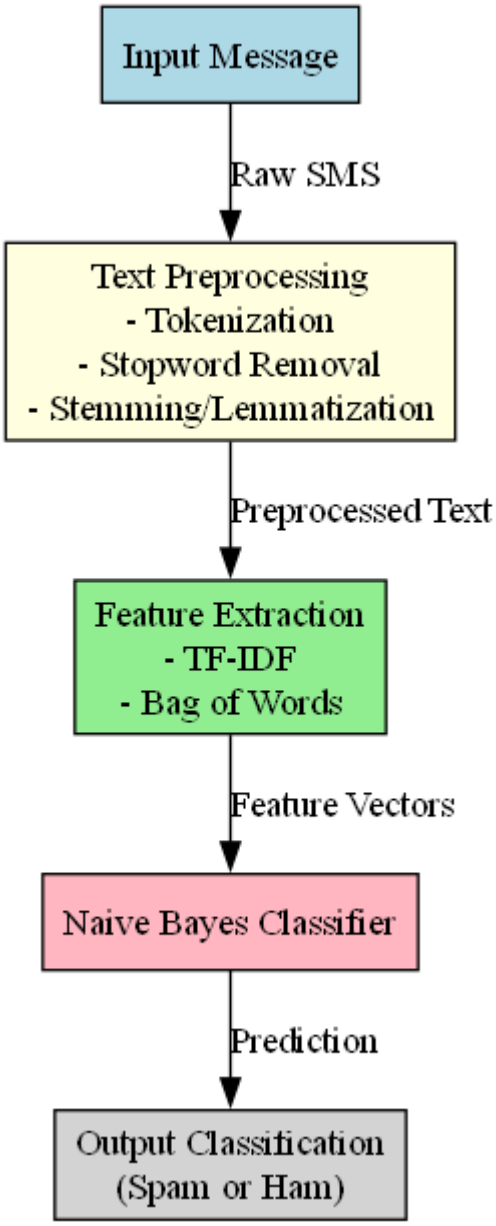


Figure 1 Flowchart of the SMS Spam Detection

The flowchart represents the end-to-end process of an SMS spam detection system that utilizes the Naive Bayes algorithm for classification. The system processes incoming SMS messages and classifies them as either **spam** or **ham** (legitimate). Here's a breakdown of each step in the flowchart:

1. Input Message

- **Description**: This is the starting point of the pipeline. The system receives an SMS message, which could be any text message from a mobile phone. The content could vary from a legitimate message (ham) to a promotional or phishing attempt (spam).
- **Data Type**: The input is raw text, often in the form of Unicode or ASCII characters.

2. Text Preprocessing

- **Description**: The received SMS message undergoes several preprocessing steps to prepare it for feature extraction and classification. These steps help standardize and clean the text, making it easier for the model to understand and analyze.

  Sub-steps:

  - **Tokenization**: The message is split into smaller units (tokens), typically words or phrases. This helps the system break down the text into manageable components.
  - **Stopword Removal**: Common words like "the," "is," or "at" are removed because they don't contribute much to identifying whether a message is spam or not. These are called "stopwords."
  - **Stemming/Lemmatization**: Words are reduced to their base or root form. For example, "running" and "ran" both become "run." This process reduces variations of the same word to a common form, aiding in feature consistency.

  Output: After preprocessing, the message is ready for the next stage — feature extraction.

3. Feature Extraction

- **Description**: The goal of this step is to convert the text data into a numerical format that a machine learning model can understand. This transformation is crucial because machine learning algorithms, like Naive Bayes, work with numerical data. Two common methods for feature extraction are:

  Sub-steps:

  - **TF-IDF (Term Frequency-Inverse Document Frequency)**: This technique calculates the importance of words in a document relative to a collection of documents. Words that appear frequently in a given message but rarely in the entire dataset are given higher weights.

- **Bag of Words (BoW)**: A simpler method that counts the occurrences of words in the text. Each word is represented by a feature, and the frequency of its appearance is recorded in the feature vector.

Output: A feature vector is produced, which represents the processed SMS message in a numerical form. This vector contains the characteristics of the message, ready to be classified.

4. Naive Bayes Classifier

- **Description**: This is the core component of the spam detection system. The Naive Bayes classifier is a probabilistic model that calculates the likelihood of a message belonging to either the "spam" or "ham" class based on the features extracted from the message. It is based on Bayes' Theorem, which calculates the conditional probability of a class given the features.

  Sub-steps:

  - The classifier is trained on a labeled dataset (messages that are already marked as spam or ham). It learns the probability distribution of words in spam and ham messages.
  - When a new message is provided, the classifier uses the features to calculate the probabilities of the message being spam or ham, based on the learned model.

  Output: A predicted probability for both classes (spam and ham). The class with the highest probability is selected as the final prediction.

5. Output Classification

- **Description**: Based on the decision made by the Naive Bayes classifier, the system outputs the final classification. The message is categorized as either:
  - **Spam**: The message is identified as promotional, fraudulent, or unwanted.
  - **Ham**: The message is identified as legitimate and non-spam.

Output: A final classification (Spam or Ham), which can be used by the system to filter messages or notify the user.

3.2 Requirement Specification

Mention the tools and technologies required to implement the solution.

3.2.1 Hardware Requirements:

- **Processor**: Multi-core CPU (e.g., Intel i5/i7 or AMD equivalent) for efficient training and evaluation of models.
- **Memory (RAM)**: Minimum 8 GB RAM for handling NLP preprocessing and machine learning tasks.
- **Storage**: At least 20 GB of free disk space to store datasets, libraries, and model checkpoints.

3.1.1 Software Requirements:

Programming Language: Python 3.8 or higher for implementing NLP preprocessing and machine learning models.

Libraries and Frameworks:

- **NLP**: NLTK for text preprocessing and tokenization.
- **Machine Learning**: Scikit-learn for ML models.
- **Data Manipulation**: Pandas and NumPy for dataset processing.
- **Visualization**: Matplotlib and Seaborn for data visualization and performance analysis.

Integrated Development Environment (IDE): PyCharm, VS Code, or Jupyter Notebook for coding and experimentation.

Operating System: Windows, macOS, or Linux (Ubuntu preferred for compatibility with machine learning frameworks).

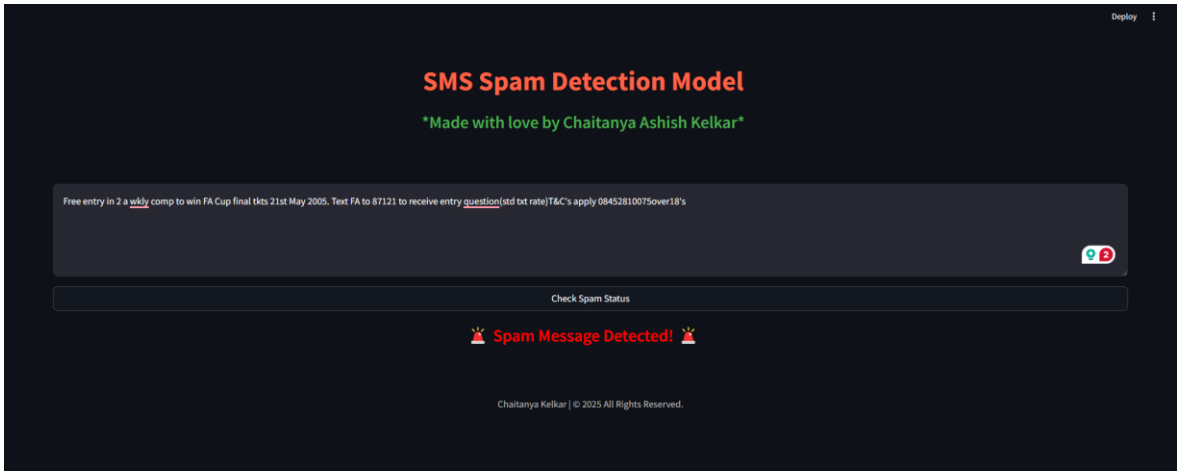
Version Control System: Git and GitHub/GitLab for tracking code changes and collaboration.

Dataset Source: Access to SMS datasets like the UCI SMS Spam Collection or Kaggle datasets for model training and evaluation.

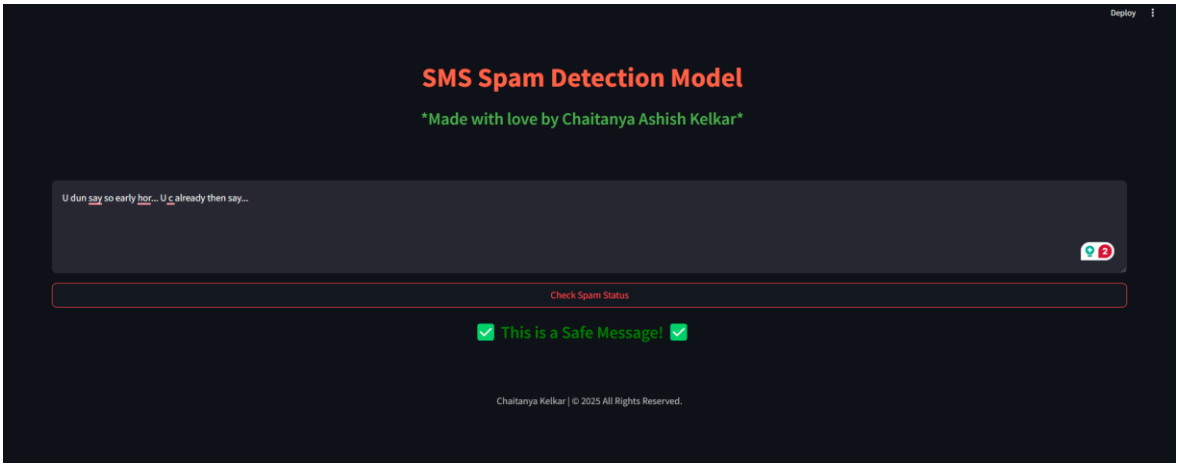
CHAPTER 4

Implementation and Result

4.1 Snap Shots of Result:



the snapshot represents the outcome of the SMS Spam Detection system, where a user inputs a message, and the system classifies it as **"Spam"**. The result, shown in a bold red font with an attention-grabbing icon (e.g., 🚨), indicates that the message contains characteristics typical of spam, such as promotional content or suspicious links. This helps the user quickly identify and avoid unwanted or potentially harmful messages.



the snapshot for **"Not Spam"** represents the outcome where the system classifies the input message as legitimate. The result, shown in a bold green font with a checkmark icon (e.g., ✅), indicates that the message is safe and does not contain characteristics

typical of spam. This reassures the user that the message is from a trusted source and is not harmful or unsolicited.

4.2 GitHub Link for Code:

https://github.com/chaitanyakelkar/SMS-Spam-Detection

CHAPTER 5

Discussion and Conclusion

5.1 Future Work:

Integration with Real-time Systems:
The current model operates on pre-processed datasets. Future enhancements can include deploying the model in real-time messaging platforms for instant spam classification and filtering.

Multilingual Support:
Extend the system to support spam detection in multiple languages by leveraging multilingual datasets and advanced language models such as mBERT or XLM-R.

Improved Feature Engineering:
Explore advanced feature extraction techniques, such as context-aware embeddings (e.g., BERT embeddings), to improve the model's ability to understand the nuances of spam messages.

Adaptive Learning:
Implement adaptive or online learning algorithms to allow the model to update dynamically as new types of spam messages emerge.

Enhanced Model Interpretability:
Develop tools and techniques to improve the interpretability of the model, allowing users to understand why a particular message was classified as spam.

Integration with Security Systems:
Combine the spam detection model with broader cybersecurity frameworks to detect phishing attempts and other malicious activities in SMS messages.

5.2 Conclusion:

The use of the Naive Bayes algorithm in this project has demonstrated its simplicity and efficiency in detecting SMS spam messages. By leveraging NLP techniques for text preprocessing and feature extraction, the model effectively classifies messages as spam or ham with high accuracy.

Despite its effectiveness, the project highlights certain limitations, such as sensitivity to imbalanced datasets and the need for advanced feature representation to handle more complex spam patterns. These challenges present opportunities for future

enhancements, such as incorporating hybrid models, adapting to multilingual text, and enabling real-time deployment.

In conclusion, the project showcases the potential of Naive Bayes as a reliable and computationally efficient solution for SMS spam detection. With further refinements and integration into real-world systems, it can contribute significantly to improving digital communication security and user experience.

REFERENCES

[1]. Ming-Hsuan Yang, David J. Kriegman, Narendra Ahuja, “Detecting Faces in Images: A Survey”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume. 24, No. 1, 2002.

[2]. V. Almeida, T. A. Almeida, and A. Yamakami, "A machine learning approach to the classification of SMS spam messages," *2013 ACM Symposium on Applied Computing*, Coimbra, Portugal, 2013, pp. 497-503.

[3]. J. Rennie, L. Shih, J. Teevan, and D. Karger, "Tackling the poor assumptions of Naive Bayes text classifiers," in *Proceedings of the 20th International Conference on Machine Learning (ICML)*, Washington, DC, USA, 2003, pp. 616-623.

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

[5]. J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[6]. R. E. Schapire, "The boosting approach to machine learning: An overview," in *Nonlinear Estimation and Classification*, Springer, 2003, pp. 149-171.

[7]. T. Joachims, "Text categorization with Support Vector Machines: Learning with many relevant features," in *Machine Learning: ECML-98*, Springer, 1998, pp. 137-142