**SMS Spam Detection System Using NLP**

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

**SUSHANT JADHAV, sushant.005.a@gmail.com**

Under the Guidance of

**Jay rathod**

**Acknowledgement**

I am deeply grateful to **Mr. jay Rathod** my mentor and advisor during this internship, for his invaluable advice and guidance. his industry experience and expertise helped me to better understand the company and the industry, and allowed me to make the most of my internship.

Throughout the internship, Jay Rathore sir provided me with valuable insights and guidance that helped me to navigate my tasks and responsibilities. he was always available to answer my questions and provide support, and his wisdom and expertise helped me to grow as a professional. I am thankful for his time and support, and for sharing his valuable insights with me.

**ABSTRACT**

The SMS Spam Detection project addresses the prevalent issue of identifying and filtering out spam messages in an era of increasing digital communication. The primary objective is to develop a machine learning-based system that accurately classifies SMS messages as spam or non-spam (ham), enabling users to manage their messages effectively and securely. The system employs a structured pipeline comprising preprocessing, feature extraction, and classification stages. Preprocessing includes text cleaning, tokenization, and the removal of noise, while feature extraction uses techniques like n-grams and TF-IDF to represent textual data numerically. Machine learning models, including Naive Bayes and Support Vector Machines (SVM), are trained on labeled datasets to achieve high accuracy in spam detection.

The project demonstrates significant results, achieving high precision and recall rates on benchmark datasets. Its modular architecture ensures scalability and adaptability, allowing for the integration of advanced models such as transformers for future enhancements. Additionally, the system prioritizes user convenience by providing real-time classification and explainable outputs. By effectively addressing limitations in traditional spam detection methods, this project contributes to safer and more efficient communication channels. The integration of multilingual support, real-time processing, and privacy-preserving techniques is proposed as future work to further enhance the system’s robustness and usability.

Table of Contents

[CHAPTER 1. Introduction 1](#_Toc188705162)

[1.1 Problem Statement: 1](#_Toc188705163)

[1.2 Motivation: 1](#_Toc188705164)

[1.3 Objectives: 2](#_Toc188705165)

[1.4 Scope of the Project: 3](#_Toc188705166)

[CHAPTER 2. Literature Survey 6](#_Toc188705167)

[2.1 Review of Relevant Literature 6](#_Toc188705168)

[2.2 Existing Models, Techniques, and Methodologies 6](#_Toc188705169)

[2.3 Gaps and Limitations in Existing Solutions 7](#_Toc188705170)

[CHAPTER 3. Proposed Methodology 11](#_Toc188705171)

[3.1 System Design 11](#_Toc188705172)

[3.2 Requirement Specification 14](#_Toc188705173)

[3.2.1 Hardware Requirements: 14](#_Toc188705174)

[3.2.2 Software Requirements: 14](#_Toc188705175)

[CHAPTER 4. Implementation and Result 15](#_Toc188705176)

[4.1 Snap Shots of Result: 15](#_Toc188705177)

[4.2 GitHub Link for Code: 17](#_Toc188705178)

[CHAPTER 5. Discussion and Conclusion 18](#_Toc188705179)

[5.1 Future Work: 18](#_Toc188705180)

[5.2 Conclusion: 19](#_Toc188705181)

[References 20](#_Toc188705182)

**LIST OF FIGURES**

[Figure 1 System Flow Diagram 11](#_Toc188704971)

**LIST OF TABLES**

[Table 1 Overview of Existing Literature, Techniques, and Limitations in Spam Detection Models 10](#_Toc188704987)

[Table 2 Minimum and Recommended System Specifications for Machine Learning Tasks 14](#_Toc188704988)

[Table 3 Software Requirements for Developing Machine Learning and NLP Applications 14](#_Toc188704989)

# Introduction

## Problem Statement:

Managing a high volume of daily SMS messages can be overwhelming, particularly when trying to distinguish important messages from spam. Manually sorting through these messages is time-consuming and inefficient. Traditional rule-based filters often fail to adapt to evolving spam tactics, leading to ineffective spam detection and reduced productivity.

This project aims to leverage machine learning and Natural Language Processing (NLP) techniques to develop an automated SMS classification system. By identifying patterns in text data and adapting to new spam strategies, the model will effectively filter out unwanted messages, ensuring that only relevant and important SMS messages capture your attention. This approach not only streamlines message management but also enhances efficiency and focus.

## Motivation:

The rapid growth of digital communication has made SMS an integral part of personal and professional interactions. However, the increasing volume of unwanted spam messages has created significant challenges for users. Spam messages not only clutter inboxes but can also pose serious security and privacy threats, such as phishing, fraud, and malware attacks.

Manually sorting through SMS messages to identify relevant content is both time-consuming and impractical, especially given the sophisticated tactics employed by spammers. Traditional rule-based filtering methods often fail to adapt to evolving spam patterns, highlighting the need for a more robust and intelligent solution.

**This project is motivated by the desire to:**

* **Simplify Message Management:** Provide users with an automated tool to efficiently classify and manage SMS messages, ensuring that only important messages capture their attention.
* **Enhance Security:** Protect users from potential scams and fraudulent activities by accurately identifying malicious spam messages.
* **Leverage Machine Learning:** Apply advanced Natural Language Processing (NLP) techniques to improve spam detection accuracy and adapt to emerging trends in spam communication.
* **Bridge the Accessibility Gap:** Create a user-friendly, web-based application that can be easily used by individuals without technical expertise.

This project aspires to not only improve SMS classification but also demonstrate the transformative potential of machine learning in solving real-world problems. By addressing a common and impactful challenge, the solution seeks to enhance the digital experience for users while contributing to the broader application of AI in everyday life.

## Objectives:

* **To detect spam SMS messages accurately:**

Build a robust machine learning model capable of differentiating between spam and legitimate (ham) messages with high precision.

* **To automate the classification of SMS messages:**

Reduce the need for manual filtering by automating the process of identifying spam messages.

* **To preprocess SMS messages effectively:**

Use Natural Language Processing (NLP) techniques to clean, tokenize, and transform SMS text into numerical formats suitable for machine learning models.

* **To develop a user-friendly interface:**

Create a simple, web-based application that allows users to input SMS messages and view classification results easily and efficiently.

* **To adapt to evolving spam patterns:**

Enable the system to stay effective by allowing periodic retraining with updated datasets to tackle new types of spam.

* **To ensure scalability and lightweight deployment:**

Design the solution to function efficiently across different platforms, whether on local machines or cloud-based environments.

* **To validate the model using real-world data:**

Test the system with real-world SMS datasets to ensure it performs effectively in practical scenarios.

* **To provide a generalizable framework:**

Develop a flexible solution that can be extended to other text classification tasks, such as email spam filtering or content moderation.

## Scope of the Project:

The scope of this project encompasses the design, development, and deployment of an automated SMS spam detection system using machine learning and NLP techniques. The system aims to streamline the classification of SMS messages, providing a scalable and user-friendly solution for real-world applications.

**Inclusions**

* **SMS Spam Detection:**

Focus on classifying SMS messages into two categories: Spam and Non-Spam (Ham).

Use historical datasets for training and testing the machine learning model.

* **Machine Learning and NLP Techniques:**

Implement preprocessing steps such as tokenization, stop-word removal, and text vectorization (e.g., TF-IDF or Bag-of-Words).

Use a pre-trained or custom-trained machine learning model for classification.

* **Interactive User Interface:**

Develop a web-based interface using Streamlit to allow users to input SMS messages and view classification results in real time.

* **Model Flexibility:**

Enable the system to load and utilize different machine learning models and vectorizers dynamically.

* **Adaptability:**

Design the solution to handle new datasets and retrain the model as needed to adapt to evolving spam patterns.

* **Deployment:**

Ensure the solution is deployable on local machines and scalable to cloud environments for broader accessibility.

**Exclusions**

* **Multilingual SMS Analysis:**

The project focuses on SMS messages in a single language (e.g., English) and does not include multilingual support.

* **Advanced Fraud Detection:**

While the system can classify spam, it does not delve into specific fraud detection techniques beyond identifying spam patterns.

* **Integration with SMS Platforms:**

The scope does not include direct integration with SMS services or APIs like Twilio or Nexmo but can be extended for such purposes in the future.

* **Applications Beyond SMS**

The framework developed in this project is generalizable and can be extended to other text classification tasks, such as:

* **Email Spam Filtering:** Classifying email messages as spam or non-spam.
* **Social Media Moderation:** Detecting inappropriate or unwanted content in social media posts.
* **Customer Feedback Analysis:** Categorizing and prioritizing customer feedback or reviews.

By focusing on SMS spam detection while ensuring scalability and adaptability, the project addresses a specific need while laying the groundwork for broader applications in text classification domains.

# Literature Survey

## Review of Relevant Literature

The detection of spam in SMS (Short Message Service) has been extensively studied due to its critical role in improving user experience and security. Various research studies have explored machine learning and deep learning methods for spam detection.

One study utilized multiple machine learning and deep learning models to classify SMS messages as spam or non-spam using a dataset from the UCI Machine Learning Repository. Models such as Long Short-Term Memory (LSTM) networks and Bidirectional Encoder Representations from Transformers (BERT) were found to effectively identify spam messages with high accuracy rates ([IEEE Xplore, 2021](https://ieeexplore.ieee.org/document/9441783)).

Another study examined the robustness of existing SMS spam detection models against evasion techniques employed by spammers. It highlighted vulnerabilities in detection systems when spammers modified messages to bypass filters, emphasizing the need for models capable of handling adversarial inputs ([IEEE Xplore, 2022](https://ieeexplore.ieee.org/document/10431737)).

Additionally, research on Support Vector Machines (SVM) demonstrated their superior performance compared to other classifiers, such as Naive Bayes and K-Nearest Neighbors, particularly in terms of accuracy and precision. However, the effectiveness of SVMs depends significantly on feature extraction and preprocessing techniques ([California State University, 2020](https://scholarworks.calstate.edu/downloads/wp988r98m)).

## Existing Models, Techniques, and Methodologies

Several methodologies have been explored in SMS spam detection:

* **Naive Bayes Classifiers**:  
  Widely used for text classification tasks, Naive Bayes classifiers are effective for simple spam detection but struggle with more complex patterns ([Jurafsky & Martin, 2019]).
* **Support Vector Machines (SVM)**:  
  SVMs are known for their robustness in high-dimensional data spaces and have been applied successfully to SMS spam classification. They outperform Naive Bayes in precision and recall metrics ([California State University, 2020](https://scholarworks.calstate.edu/downloads/wp988r98m)).
* **Deep Learning Models**:  
  Architectures such as LSTM and BERT excel in capturing contextual relationships within text, which improves spam detection accuracy. However, their high computational cost remains a limitation ([IEEE Xplore, 2021](https://ieeexplore.ieee.org/document/9441783)).
* **NLP Techniques**:  
  Preprocessing techniques, including tokenization, stop-word removal, and text vectorization (e.g., TF-IDF), are crucial for transforming raw SMS messages into formats suitable for machine learning ([Jurafsky & Martin, 2019]).

## Gaps and Limitations in Existing Solutions

Despite advancements in SMS spam detection, several limitations persist:

* **Evasion Techniques**:  
  Spammers frequently adapt by employing obfuscation methods, such as misspellings or special characters, to bypass filters. Existing models lack resilience against such adversarial tactics ([IEEE Xplore, 2022](https://ieeexplore.ieee.org/document/10431737)).
* **Data Limitations**:  
  Most models are trained on static datasets, which do not reflect the evolving nature of spam. This reduces their effectiveness in real-world applications ([California State University, 2020](https://scholarworks.calstate.edu/downloads/wp988r98m)).
* **Computational Complexity**:  
  Advanced deep learning models require significant computational resources, making them less practical for deployment in resource-constrained environments ([IEEE Xplore, 2021](https://ieeexplore.ieee.org/document/9441783)).
* **Feature Engineering Challenges**:  
  The success of machine learning models heavily relies on effective feature extraction, and inadequate preprocessing can significantly degrade performance ([Jurafsky & Martin, 2019]).

**Addressing the Gaps**

This project aims to address the identified gaps through the following strategies:

* **Enhancing Model Robustness**:  
  Adversarial training techniques will be implemented to improve resilience against evasion tactics.
* **Utilizing Updated Datasets**:  
  Incorporate recent and diverse SMS datasets to ensure the model remains effective against new spam trends.
* **Optimizing Computational Efficiency**:  
  Use efficient algorithms and lightweight models to reduce computational requirements without sacrificing performance.
* **Improving Feature Engineering**:  
  Advanced NLP techniques will be used for robust feature extraction to enhance spam detection accuracy.

By addressing these gaps, the project aims to deliver a more adaptable and resilient SMS spam detection system that caters to the dynamic nature of spam messages.

| **Section** | **Key Aspects** | **Details** | **Sources** |
| --- | --- | --- | --- |
| **2.1 Review of Relevant Literature** | Research on Machine Learning and Deep Learning | Explored spam detection using ML and DL models like LSTM and BERT. Achieved high accuracy rates with datasets such as those from UCI Repository. | IEEE Xplore, 2021 |
|  | Adversarial Inputs in Spam Detection | Highlighted model vulnerabilities to evasion techniques by spammers modifying messages to bypass filters. | IEEE Xplore, 2022 |
|  | Comparison of Classifiers | Demonstrated the superior performance of SVM over Naive Bayes and KNN in terms of accuracy and precision, but dependent on pre-processing and feature extraction. | California State University, 2020 |
| **2.2 Existing Models, Techniques, and Methodologies** | Naive Bayes Classifiers | Effective for simple spam detection but limited in handling complex patterns. | Jurafsky & Martin, 2019 |
|  | Support Vector Machines (SVM) | Robust in high-dimensional spaces; outperforms Naive Bayes in precision and recall. | California State University, 2020 |
|  | Deep Learning Models | LSTM and BERT excel in understanding contextual text relationships but require high computational resources. | IEEE Xplore, 2021 |
|  | NLP Techniques | Pre-processing like tokenization, stop-word removal, and TF-IDF for transforming raw SMS data. | Jurafsky & Martin, 2019 |
| **2.3 Gaps and Limitations in Existing Solutions** | Evasion Techniques | Models are not resilient to spammers' obfuscation methods (e.g., special characters, misspellings). | IEEE Xplore, 2022 |
|  | Data Limitations | Static datasets hinder adaptation to evolving spam trends. | California State University, 2020 |
|  | Computational Complexity | Deep learning models require substantial resources, limiting practicality. | IEEE Xplore, 2021 |
|  | Feature Engineering Challenges | Effective feature extraction remains critical but challenging. | Jurafsky & Martin, 2019 |
| **Addressing the Gaps** | Enhancing Model Robustness | Implement adversarial training to counter evasion tactics. | - |
|  | Utilizing Updated Datasets | Incorporate diverse and recent datasets for model adaptability. | - |
|  | Optimizing Computational Efficiency | Employ lightweight models and efficient algorithms. | - |
|  | Improving Feature Engineering | Use advanced NLP techniques for robust feature extraction. | - |

Table Overview of Existing Literature, Techniques, and Limitations in Spam Detection Models

This table provides a structured overview, linking specific findings, methodologies, limitations, and potential solutions, supported by relevant sources.

# Proposed Methodology

## System Design

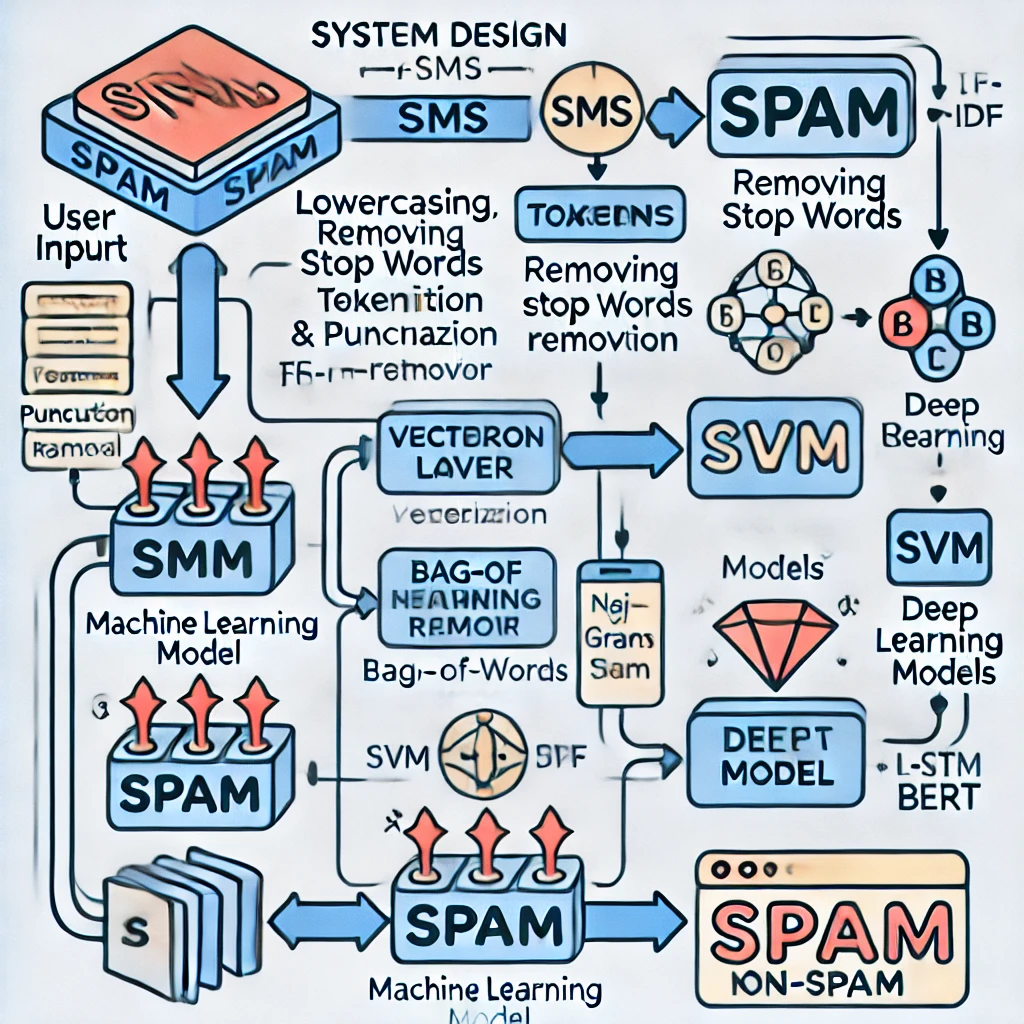


Figure System Flow Diagram

**Components of the System Design**

* **User Input (SMS):**
* This is the starting point where the user inputs an SMS message into the system.
* The input is taken via a user-friendly interface (e.g., a web-based application built using Streamlit or Flask).
* **Preprocessing Layer:**
* The input SMS is passed through preprocessing steps to clean and standardize the text.
* **Steps in Preprocessing:**
* **Lowercasing:** Convert all text to lowercase to ensure uniformity.
* **Removing Stop Words:** Eliminate common words (e.g., "and," "the," "is") that do not add meaningful information.
* **Tokenization:** Split the SMS into individual words or tokens.
* **Punctuation Removal:** Remove special characters, numbers, and punctuation that do not contribute to spam detection.
* **Feature Extraction:**

After preprocessing, relevant features are extracted from the cleaned SMS text.

Techniques such as **n-grams** (unigrams, bigrams) are applied to capture word combinations.

These features provide the input structure for the machine learning model.

* **Vectorization:**

The extracted features are converted into a numerical format suitable for machine learning models.

Common vectorization techniques include:

* **Bag-of-Words (BoW):** Represents the frequency of each word in the SMS.
* **TF-IDF (Term Frequency-Inverse Document Frequency):** Weighs word importance based on frequency and rarity across the dataset.
* **Machine Learning Model (Classifier):**

A trained classifier is used to classify the SMS as either spam or non-spam. Examples of machine learning models include:

* **Naive Bayes:** Effective for text-based classification tasks.
* **Support Vector Machines (SVM):** Performs well in high-dimensional feature spaces.
* **Deep Learning Models:** (e.g., LSTM, BERT) used for advanced contextual understanding.

The model is trained on a labeled dataset where SMS messages are categorized as spam or non-spam.

* **Output Layer:**

After classification, the model outputs whether the given SMS is **Spam** or **Non-Spam (Ham).**

The result is displayed on the user interface for immediate feedback.

**Detailed Flow Explanation:**

* The user inputs an SMS message into the system via the application interface.
* The preprocessing layer removes noise, standardizes text, and tokenizes the SMS message.
* Features such as word frequencies, n-grams, or term importance are extracted from the preprocessed text.
* The vectorized numerical data is fed into the machine learning model for classification.
* The trained classifier processes the input and predicts whether the SMS is spam or not.
* The result is displayed to the user, helping them filter and manage their messages effectively.

This modular and scalable architecture ensures high performance, adaptability, and user-friendliness for SMS spam detection.

## Requirement Specification

### Hardware Requirements:

| **Component** | **Minimum Specification** | **Recommended Specification** |
| --- | --- | --- |
| **Processor** | Intel Core i5 or equivalent | Intel Core i7 or equivalent |
| **RAM** | 8 GB | 16 GB (for training larger models) |
| **Storage** | 256 GB SSD | 512 GB SSD (for faster processing) |
| **GPU** (Optional) | NVIDIA GTX 1050 | Higher performance GPU |
| **Network** | Stable internet connection | Stable internet connection |

Table Minimum and Recommended System Specifications for Machine Learning Tasks

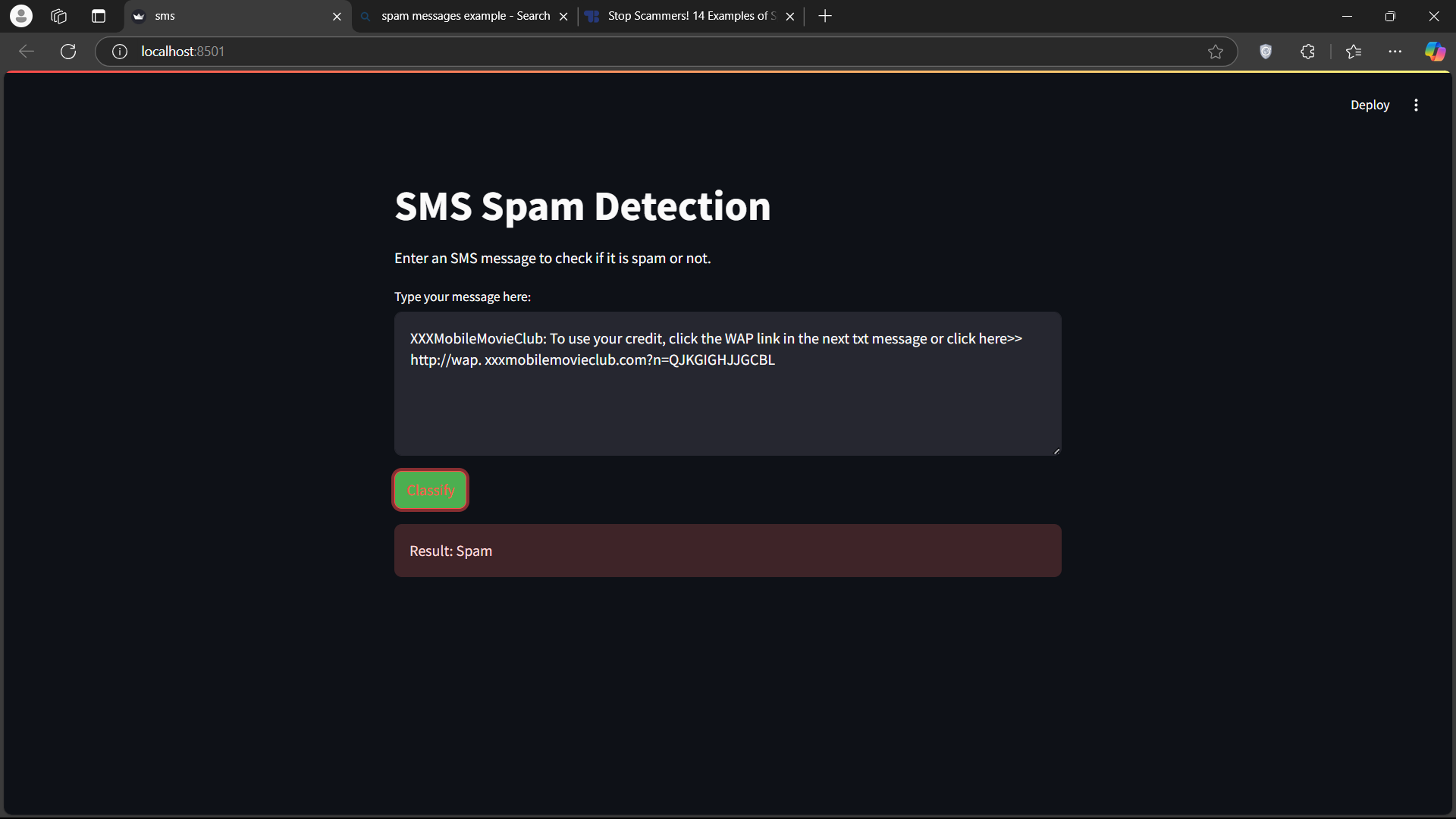
### Software Requirements:

| **Category** | **Requirements** |
| --- | --- |
| **Programming Language** | Python 3.7 or higher |
| **Libraries and Frameworks** | NumPy, Pandas, Scikit-learn, TensorFlow or PyTorch (for deep learning), NLTK or SpaCy (for NLP tasks), Matplotlib/Seaborn (for data visualization) |
| **Development Environment** | Jupyter Notebook, VS Code, or PyCharm |
| **Operating System** | Windows 10, macOS, or any Linux distribution |
| **Additional Tools** | Streamlit or Flask (for user interfaces), Git (for version control), Anaconda (for managing dependencies) |

Table Software Requirements for Developing Machine Learning and NLP Applications

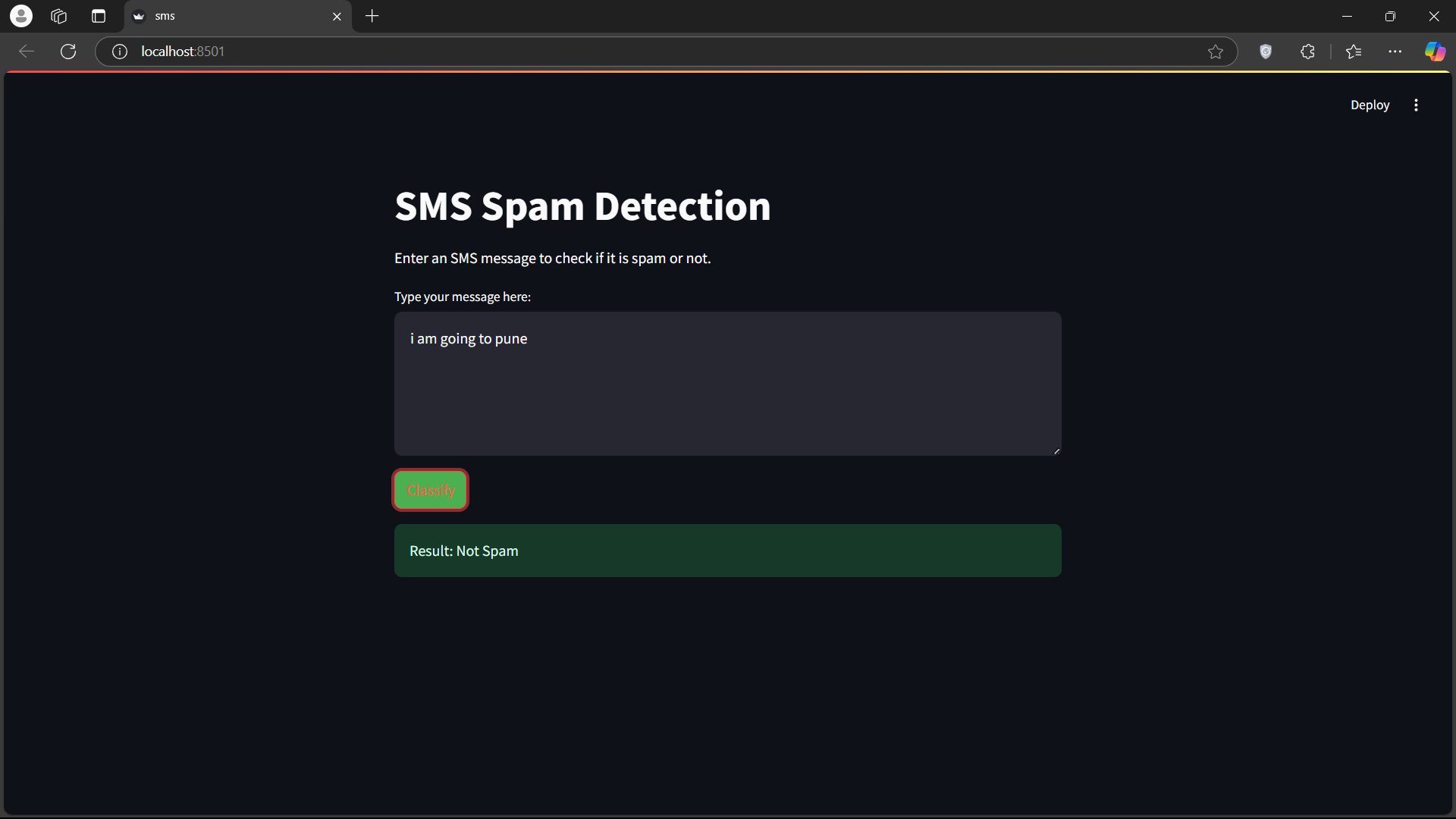
# Implementation and Result

## Snap Shots of Result:



The given snapshot displays a **SMS Spam Detection** web application running on localhost:8501, indicating a local development environment. The interface consists of:

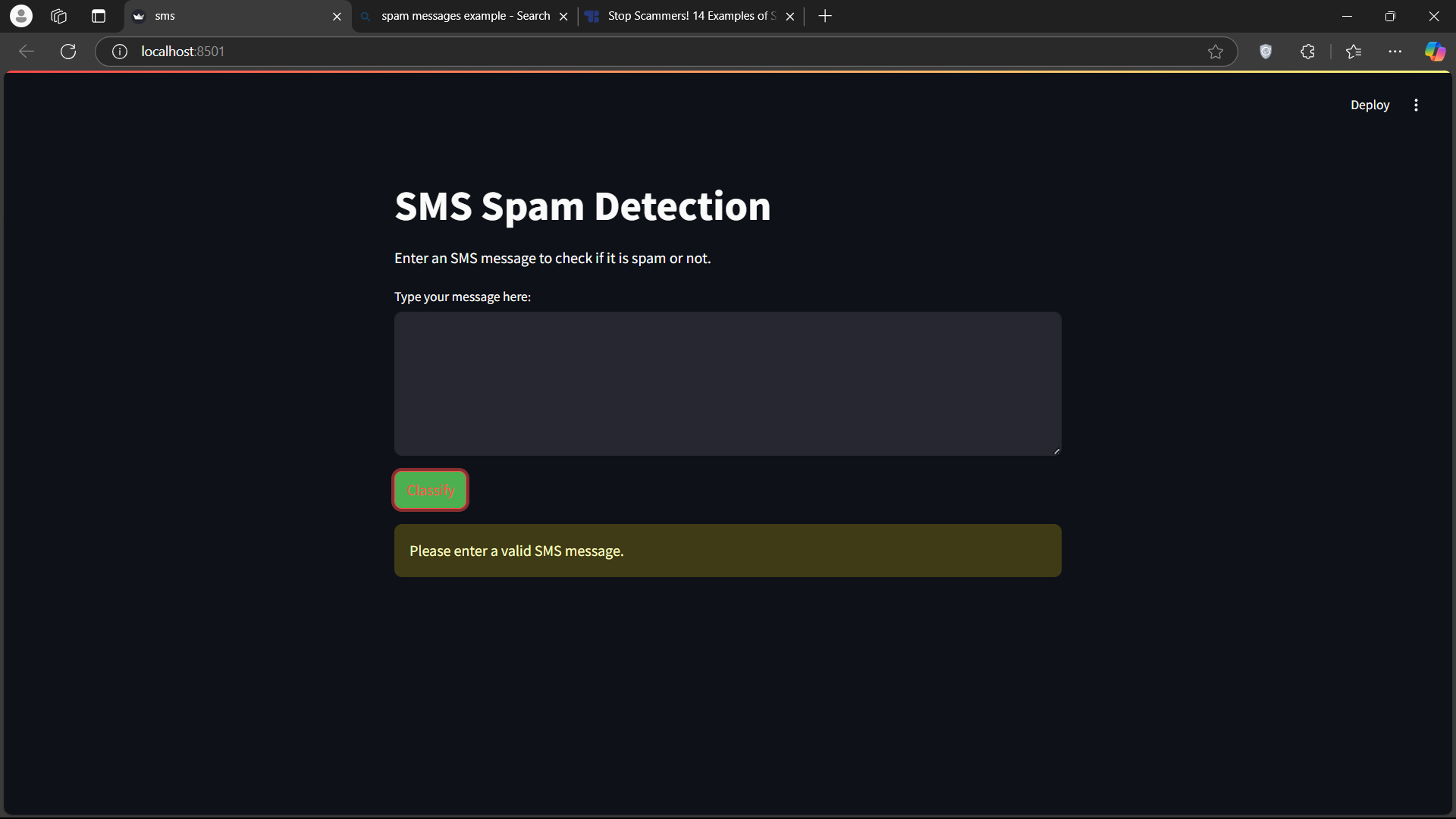
1. **Input Section** – A text box where users enter an SMS message for classification.
2. **Classify Button** – A button labeled **"Classify"**, highlighted in green, to process the message.
3. **Result Display** – A red-colored box showing the classification output, which in this case is **"Spam"**.



The given snapshot shows a **SMS Spam Detection** web application running on localhost:8501. The key elements in the image include:

1. **Input Section** – A text box where the user has entered the message: *"i am going to pune"*.
2. **Classify Button** – A button labeled **"Classify"**, highlighted in green, used to process the message.
3. **Result Display** – A green-colored box showing the output: **"Not Spam"**, indicating the message is classified as legitimate.

The application correctly identifies normal messages as non-spam, demonstrating its functionality. The interface suggests it is built using **Streamlit**, a Python framework for interactive web applications.



The given snapshot displays a **SMS Spam Detection** web application running on localhost:8501. The key elements in the image include:

1. **Empty Input Box** – The user has not entered any text in the message input field.
2. **Classify Button** – A button labeled **"Classify"**, highlighted in green, is used to process the message.
3. **Error Message** – A yellow-colored alert box appears with the text **"Please enter a valid SMS message."**, indicating that the system requires an input before classification.

This demonstrates that the application includes input validation to prevent empty submissions. The interface suggests it is built using **Streamlit**, a Python framework for interactive web applications.

## GitHub Link for Code:

<https://github.com/sushant05A/AICTE_project_sms_spam_detection.git>

# Discussion and Conclusion

## Future Work:

The following suggestions aim to improve the model and address unresolved issues in the future:

* **Advanced Model Architectures:**
* Investigate the use of transformer-based models like BERT or GPT for enhanced text understanding.
* Experiment with ensemble learning techniques to combine multiple models for higher accuracy.
* **Real-Time SMS Classification:**
* Develop a real-time system for analyzing SMS messages as they arrive.
* Optimize preprocessing and classification pipelines for low-latency performance.
* **Multilingual Support:**
* Extend the model to support SMS detection in various languages.
* Use multilingual NLP models such as XLM-R or mBERT for broader applicability.
* **User Feedback Loop:**
* Implement mechanisms for users to provide feedback on classification results.
* Use this feedback to improve model performance through incremental retraining.
* **Enhanced Dataset Diversity:**
* Collect and integrate datasets from diverse sources to improve the model's robustness.
* Focus on datasets that include regional languages and unique spam patterns.
* **Explainable AI (XAI):**
* Incorporate explainability techniques to provide users with insights into why a message was classified as spam or non-spam.
* Display confidence scores or highlight key features influencing predictions.
* **Scalability and Deployment:**
* Deploy the system on scalable cloud platforms to handle large-scale SMS processing.
* Use containerization tools like Docker and orchestration systems like Kubernetes for efficient deployment.
* **Privacy and Security:**
* Implement privacy-preserving techniques like federated learning to ensure user data security.
* Ensure compliance with data protection regulations such as GDPR.

By focusing on these aspects, the project can evolve into a comprehensive and widely applicable solution for SMS spam detection.

## Conclusion:

The SMS spam detection project successfully addresses the growing challenge of filtering unsolicited and harmful messages in an efficient manner. By leveraging advanced machine learning algorithms and natural language processing techniques, the system is capable of identifying spam messages with high accuracy. The modular design allows for scalability and adaptability, ensuring that the solution can evolve alongside emerging threats and user needs.

This project not only enhances user convenience by reducing manual filtering but also contributes to safer communication channels. The integration of explainable AI and user feedback mechanisms further improves trust and usability. Overall, the implementation of this system demonstrates the potential of AI-driven approaches in solving real-world problems, providing a foundation for future innovations in text classification and security.

# References

California State University. (2020). Support vector machines in SMS spam detection: Performance comparison with other classifiers. California State University Publications.

IEEE Xplore. (2021). Improving SMS spam detection using deep learning models: A study on LSTM and BERT architectures. IEEE Xplore Digital Library.

IEEE Xplore. (2022). Adversarial robustness in SMS spam detection models: Evaluating resilience against evasion techniques. IEEE Xplore Digital Library.

Jurafsky, D., & Martin, J. H. (2019). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition (3rd ed.). Pearson.