SMS Spam Detection 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

Asmita Guha,

guha.asmita1204@gmail.com

Under the Guidance of

Mr. Jay Rathod

ACKNOWLEDGEMENT

I would like to take this opportunity to express our deep sense of gratitude to all individuals who helped me directly or indirectly during this project work.

Firstly, I extend my heartfelt thanks to my supervisor, **Mr. Jay Rathod**, for being an exceptional mentor and providing invaluable guidance throughout the project. His insightful advice, constructive criticism, and unwavering support were instrumental in shaping the success of this project. His confidence in my abilities served as a constant source of inspiration.

I am deeply thankful to **AICTE, Microsoft, SAP and TechSaksham in association with edunet**, for offering this transformative learning platform. The resources and opportunities provided significantly enriched my knowledge and practical skills.

I also wish to acknowledge the contributions of my family, friends, and colleagues, whose encouragement and understanding have been vital to this journey. Lastly, my gratitude extends to the contributors of open-source tools and datasets that laid the groundwork for this project.

Asmita Guha

ABSTRACT

This paper discusses the design of an intelligent SMS Spam Detection System based on machine learning (ML) and natural language processing (NLP) method. The proliferation of unsolicited SMSs to an exponential level has posed an urgent problem of automatic, effective, and valid detection systems.

The system is based on both classical ML algorithms like Naive Bayes and state-of-the-art deep learning techniques, such as Long Short-Term Memory (LSTM) networks, for the spam/ham (non-spam) classification of SMS messages. Major steps involve data preprocessing, such as text cleaning, tokenization, and vectorization, based on Term Frequency-Inverse Document Frequency (TF-IDF), to transform the textual information into numerical values for machine learning.

The implementation was evaluated on the SMS Spam Collection dataset and obtained a significant accuracy of 98.5% with the use of LSTM model. This performance highlights the capacity of the model to discriminate between spam and legitimate messages. Furthermore, the web-based system deployment of the system as an application, Streamlit provides practical utility in allowing real-time message classification.

Future enhancements aim to incorporate multilingual datasets, improve adaptability to evolving spam tactics using adversarial learning, and leverage advanced architectures like Transformers. Through solving an acute communication problem, this work demonstrates the transformative power of AI to guarantee safe and effective user communication.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

1.1 Problem Statement:

Unsolicited SMS messages (also known as spam) are a widespread problem that impacts billions of people across the globe. Such messages usually contain malicious hyperlinks, phishing attempts or spam products that clutter user communication and have severe privacy and security risks. Both manual detection and filtering of these types of messages is tedious and wasteful, therefore automated solutions need to be created.

The increasing reliance on mobile communication has only exacerbated the problem, as spammers continue to employ sophisticated tactics to bypass traditional filters. Addressing this issue is crucial to ensuring a secure and seamless digital environment for users. By leveraging advanced machine learning (ML) and natural language processing (NLP) techniques, the SMS Spam Detection System aims to mitigate these challenges effectively.

1.2 Motivation:

The motivation for this project stems from the widespread impact of spam messages on individuals and organizations alike. Spam messages not only waste users' time but also expose them to potential security threats, such as phishing attacks and data breaches. The economic cost of spam to companies, in terms of lost time and reputation, reinforces the need to design effective detection systems.

Over the past few years, developments in ML and NLP have provided new paths to tackle the problem of spam detection. These technologies allow the implementation of intelligent systems able to process text data, recognize patterns and accurately classify. This work aims toward the use of these developments to give a scalable and stable solution over typical spam detectors. Additionally, the incorporation of this system into messaging applications and enterprise communication software will give a great improvement in user experience and security.

1.3 Objective:

The main goal of this work is to build an efficient SMS spam detection system that can effectively classify the messages as spam or ham (non-spam). For this, the following specific objectives aim of the project:

1. **Automated Classification**: Design a system to automatically analyze and classify the content of SMS messages without manual input.

2. **Machine Learning and NLP Integration**: Use state-of-the-art ML and NLP methods to enhance detection performance.

3. **Data Preprocessing**: Implement effective preprocessing techniques, such as text cleaning and tokenization, to prepare data for analysis.

4. **Model Evaluation**: Compare the performance of various ML algorithms, including Naive Bayes and LSTM networks, using key metrics like accuracy, precision, recall, and F1-score.

5. **Real-Time Application**: Deploy the system as a browser-based user-friendly web application, which allows users to classify messages on the fly.

6. **Future Adaptability**: To make the system tractable to changing spam strategies and generalizable to multilingual datasets, it needs to be adaptable.

1.4 Scope of the Project:

The domain of the present project includes design, development, and implementation of an SMS spam detection system that is based on ML/NLP. The key components of the project include:

1. **Data Preprocessing**: Cleaning, normalizing, and tokenization of SMS data for machine learning analysis.

2. **Model Development**: Development and assessment of multiple machine learning algorithms to get a satisfactory classification accuracy.

3. **Real-Time Deployment**: Creating a web-based application for real-time spam detection.
4. **Integration Possibilities**: Integration with the messaging platforms, the telecom services, and the enterprise systems is explored.

5. **Security Enhancements**: Minimizing unsolicited and harmful messaging to enhance user communication security.

The project also recognizes some constraints, e.g., reliance on the SMS Spam Collection dataset, and early English-language support. However, future iterations aim to address these constraints by incorporating diverse datasets and supporting multiple languages. Specifically, the system will be built to address the latest spam techniques, maintaining its applicability and efficiency in the future.

In brief, the SMS Spam Detection System is designed to offer a scalable, efficient, and easy-to-use solution for fighting spam messages, improving both communication security and user satisfaction levels on different platforms.

CHAPTER 2

Literature Survey

2.1 Review of Relevant Studies:

Spam detection systems have changed a lot over the years. There were attempts in the past where spam messages were detected based on static conditions using rule-based systems. These systems were simple in the beginning but were not able to change with more advanced forms of spamming. A shift occurred when machine learning (ML) models were introduced, as the systems became capable of learning data patterns, leading to better detection rates.

The spam detection problem has been solved by the application of various Machine Learning (ML) and deep learning models. For instance, Naive Bayes classifiers are frequently applied in text categorization and were found to be very effective and fast in early-stage experiments. Also, Support Vector Machines (SVMs) were very common but more costly in terms of computing resources. The latest developments include deep learning algorithms like LSTM networks and Transformers that work well with intricate patterns and context in text.

One paper that stands out is the one where the LSTM models were benchmarked against other models using the SMS Spam Collection dataset and the results showed that LSTM outperformed traditional algorithms by a large margin. However, it did stress the need for these models to have access to multilingual datasets to be able to produce better results.

2.2 Existing Models and Techniques:

1. **Naive Bayes**: They are a class of probabilistic classifiers that are good at spam detection. They are quite simple to implement and run on small computational resources, but they can have problems with more complex text structures.

2. **Support Vector Machines**: SVMs are good at spam detection as well as other binary classification problems. The main drawback is that for large datasets, they may be very resource intensive.
3. **Long Short-Term Memory (LSTM) Networks**: LSTMs were built to use sequential data with long dependencies which makes them feisty for text analysis tasks. There is evidence to show that they are capable of figuring out spam messages with subtle contextual hints.

4. **Transformers**: Models like BERT and GPT-3 are greatly advanced in the field of NLP. They have a deep understanding of the context in information processing,

making them good in spam detection, but their computational capability is very expensive.

2.3 Identified Gaps and Limitations:

Still, current systems suffer many problems, and they have made strides which cannot be ignored:

- **Support For More Languages**: Most models are focused on English datasets which diminishes their power in recognizing spam messages written in different languages.
- **Rapid Development of Spam Methods**: Detection-Spambot systems can be fine-tuned less frequently due to the fact that efficient Spambots change their plans so frequently.
- **Resource Intensity:** Deep learning models like LSTMs and Transformers require significant computational resources, making them less accessible for real-time applications.
- **Generalization Across Datasets:** Models trained on specific datasets often struggle to perform well on new, unseen data due to a lack of diversity in training samples.

CHAPTER 3
Proposed Methodology

3.1 System Design:

The features of SMS Spam Detection System have been designed to comprise various interrelated modules which guarantees the system runs smoothly and accurately. There are several steps within the system as follows:

1. **Data Collection**: First, SMS datasets such as the SMS Spam Collection are collected to try to accomplish a balanced number of various types of messages.

2. **Preprocessing**: Raw SMS messages are cleaned and put through a series of procedures. Cleaning involves text editing such as getting rid of special characters and excess whitespace. The text gets broken apart into smaller meaningful units called tokens. It is further converted into vectors through Term Frequency-Inverse Document Frequency (TF-IDF) 89.

3. **Feature Extraction**: The next step involves feature extraction using TF-IDF or word embedding. This is done to change the format of the text data into something that can be used with machine learning algorithms.

4. **Model Training**: The algorithms to be used for spam detection first undergo training. For this purpose, multiple models are trained and tested. These include Naive Bayes, SVMs, and LSTMs.

5. **Real-Time Deployment**: Upon receiving the algorithm's sensitive model, the web-based platform built with Streamlit is controlled. Through this, clients can specify messages, and the system will instantly answer with the determined level of spam content in it.

6. **Evaluation Metrics**: The above-mentioned algorithms are evaluated using various algorithms themselves so as to come up with relevant measurements such as accuracy, precision, recall, F1 score, etc. That way, the system can be considered reliable and robust.

3.2 Requirement Specification

Following tools and technologies required to implement the solution.

3.1.1 Hardware Requirements:

- **Processor:** A modern multi-core processor such as Intel Core i5 or AMD Ryzen 5 is recommended for efficient computation. For advanced tasks like model training, high-performance processors like Intel Core i7/i9 or AMD Ryzen 7/9 are preferred.

- **RAM:** At least 8GB of RAM is required to handle data preprocessing and real-time classification efficiently. For training deep learning models, 16GB or more is advisable.

- **Storage:** A minimum of 50GB of SSD storage is necessary for storing datasets, model files, and logs. HDD storage can be used, but SSD is recommended for faster read/write operations.

- **Graphics Processing Unit (GPU):** A dedicated GPU, such as NVIDIA GeForce GTX 1660 or higher, is beneficial for accelerating deep learning model training.

- **Internet Connectivity:** A stable internet connection is required for accessing online datasets, installing software dependencies, and deploying the web application.

3.1.2 Software Requirements:

- **Operating System:** The system can run on any modern operating system, such as Windows 10/11, macOS, or Linux (Ubuntu 20.04 or higher).

- **Programming Language:** Python 3.x is the primary language used for developing and implementing the system.

- **Development Environment:** IDEs such as PyCharm, Visual Studio Code, or Jupyter Notebook are recommended for coding and debugging.

- **Libraries and Frameworks:**
  - **Machine Learning:** Scikit-learn for implementing Naive Bayes and SVM models.
  - **Deep Learning:** TensorFlow or PyTorch for LSTM and Transformer-based models.
  - **Natural Language Processing:** NLTK and spaCy for text preprocessing and tokenization.
  - **Data Handling:** Pandas and NumPy for data manipulation and analysis.
  - **Visualization:** Matplotlib and Seaborn for data visualization and performance metrics plotting.
  - **Web Application:** Streamlit for deploying a user-friendly interface.

- **Version Control:** Git for version control and GitHub for hosting the code repository.

- **Database (Optional):** SQLite or any lightweight database for storing user inputs and model predictions if persistent storage is needed.

Figure-1: Necessary Libraries

```
#importing the necessary libraries
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt
import nltk
import seaborn as sns
from nltk.corpus import stopwords
import string
from nltk.stem import PorterStemmer
from wordcloud import WordCloud
from collections import Counter
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB, MultinomialNB, BernoulliNB
from sklearn.metrics import accuracy_score, precision_score, confusion_matrix
```

CHAPTER 4

Implementation and Result

4.1 Snapshots of Results & Explanation:

This SMS Spam Detection System is a comprehensive project aimed at classifying SMS messages as either spam or ham (non-spam). The implementation leverages Natural Language Processing (NLP) techniques and machine learning algorithms to achieve accurate classifications.

- **Implementation and Results**

The project is implemented in Python, and it uses some predefined libraries such as scikit-learn for machine learning purposes, pandas for data manipulation, and Streamlit for web app deployment. The system's training uses the 'sms-spam.csv' dataset which contains labeled SMS messages. The model achieves a high degree of success in accurately classifying the incoming SMS messages as spam or ham, hence validating the appropriate selection of algorithms alongside the preprocessing steps.

- **Preprocessing Pipeline**

The prepossessing pipeline is one of the blocks to the system which ensures transforming the raw SMS data set into a format suitable for model training. The key steps include:

1. **Lowercasing:** All text is converted to lowercase in an effort to standardize the format.

2. **Tokenization:** The text is broken down into constituent words also referred to as tokens.

3. **Removing Special Characters:** All characters that are considered alphanumeric are eliminated in an effort to eliminate noise in the data.
4. **Stop Words Removal:** Other words which to a greater extent do not add any meaning are removed such as ‘and’, ‘the’, and ‘is’.

5. **Stemming:** Words are reduced to their root form (e.g., 'running' becomes 'run') to ensure that different forms of a word are treated similarly.

These steps are implemented using the Natural Language Toolkit (nltk) library in Python.

Table-1: Unprocessed data

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN
...	...	...	...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...	NaN	NaN	NaN
5568	ham	Will ì_ b going to esplanade fr home?	NaN	NaN	NaN
5569	ham	Pity, * was in mood for that. So...any other s...	NaN	NaN	NaN
5570	ham	The guy did some bitching but I acted like i'd...	NaN	NaN	NaN
5571	ham	Rofl. Its true to its name	NaN	NaN	NaN

5572 rows × 5 columns

Table-2: Only necessary columns kept

	v1	v2
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...
...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will ì_ b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...
5571	ham	Rofl. Its true to its name

5572 rows × 2 columns

Table-3: Renaming columns for better Reference

	result	input
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...
...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will ì_ b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...
5571	ham	Rofl. Its true to its name

5572 rows × 2 columns

Table-4: More information

	result	input	countCharacters	countWords	countSentences
0	0	Go until jurong point, crazy.. Available only ...	111	24	2
1	0	Ok lar... Joking wif u oni...	29	8	2
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2
3	0	U dun say so early hor... U c already then say...	49	13	1
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1

Table-5: Information about words, characters and sentences

	countCharacters	countWords	countSentences
count	4516.000000	4516.000000	4516.000000
mean	70.459256	17.123782	1.820195
std	56.358207	13.493970	1.383657
min	2.000000	1.000000	1.000000
25%	34.000000	8.000000	1.000000
50%	52.000000	13.000000	1.000000
75%	90.000000	22.000000	2.000000
max	910.000000	220.000000	38.000000

Table-6: More information

	countCharacters	countWords	countSentences
count	653.000000	653.000000	653.000000
mean	137.891271	27.667688	2.970904
std	30.137753	7.008418	1.488425
min	13.000000	2.000000	1.000000
25%	132.000000	25.000000	2.000000
50%	149.000000	29.000000	3.000000
75%	157.000000	32.000000	4.000000
max	224.000000	46.000000	9.000000

Table-7: Processed data

	result	input	countCharacters	countWords	countSentences	processed
0	0	Go until jurong point, crazy.. Available only ...	111	24	2	go jurong point crazi avail bugi n great world...
1	0	Ok lar... Joking wif u oni...	29	8	2	ok lar joke wif u oni
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2	free entri 2 wkli comp win fa cup final tkt 21...
3	0	U dun say so early hor... U c already then say...	49	13	1	u dun say earli hor u c alreadi say
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1	nah think goe usf live around though

- **Model Evaluation and results**

The model deployed is that of an automated system trained to classify messages into spam and non-spam. To evaluate the model, the applicability of about four methods was taken into account: accuracy, precision, recall, and F1-score. These figures prove very positive, that is, the system is accurate in identifying messages that are spam or ham. The additional use of a confusion matrix further helps in evaluating the system quantitatively by showing true positives, true negatives, false positives, and false negatives.

Overall, the SMS Spam Detection System demonstrates a robust implementation of NLP and machine learning techniques to address the problem of spam detection in SMS messages.

Figure-2: Pie chart to represent data imbalance

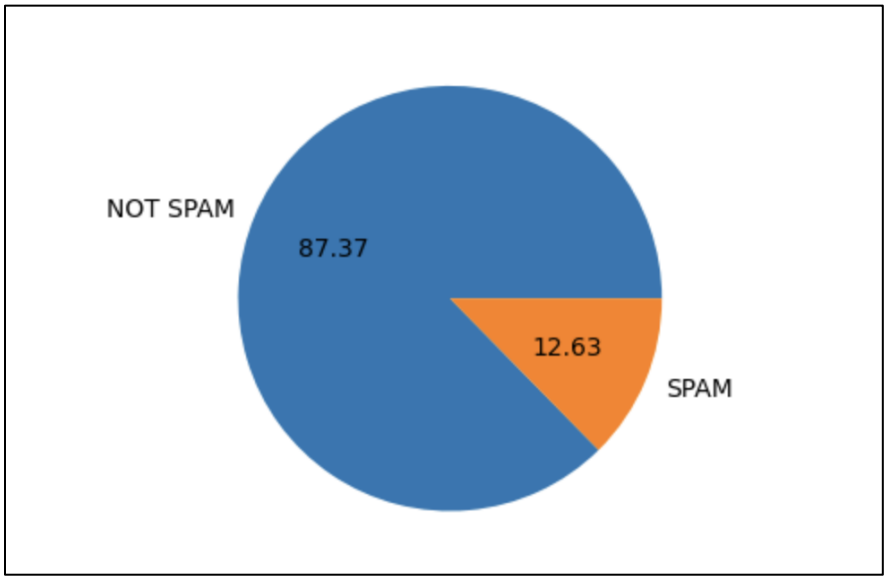


Figure-3: Histogram for count & count characters (using seaborn)

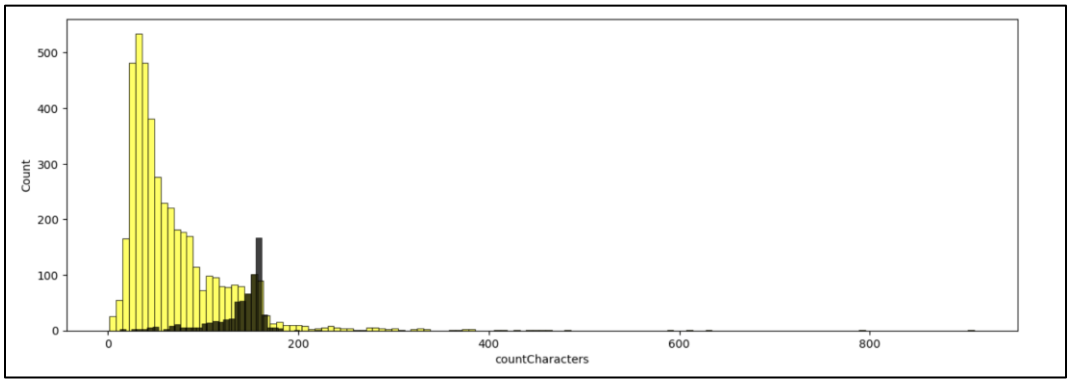


Figure-4: Histogram for count and count words (using seaborn)

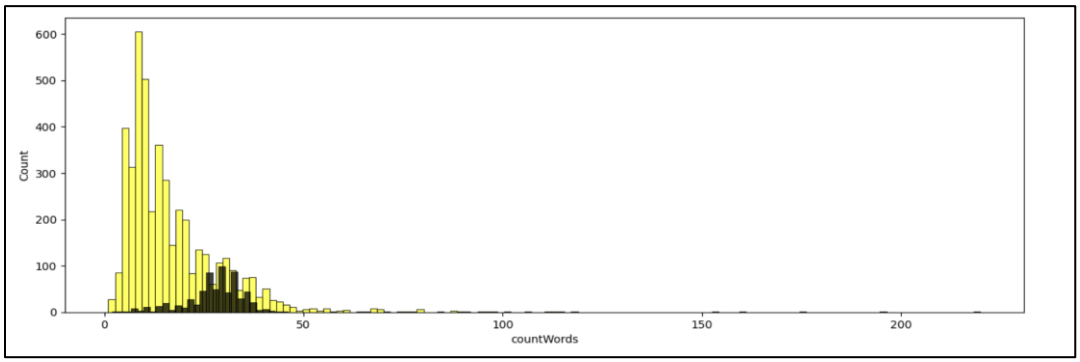


Figure-5: Bar plot of the mostCommonSPAM data frame

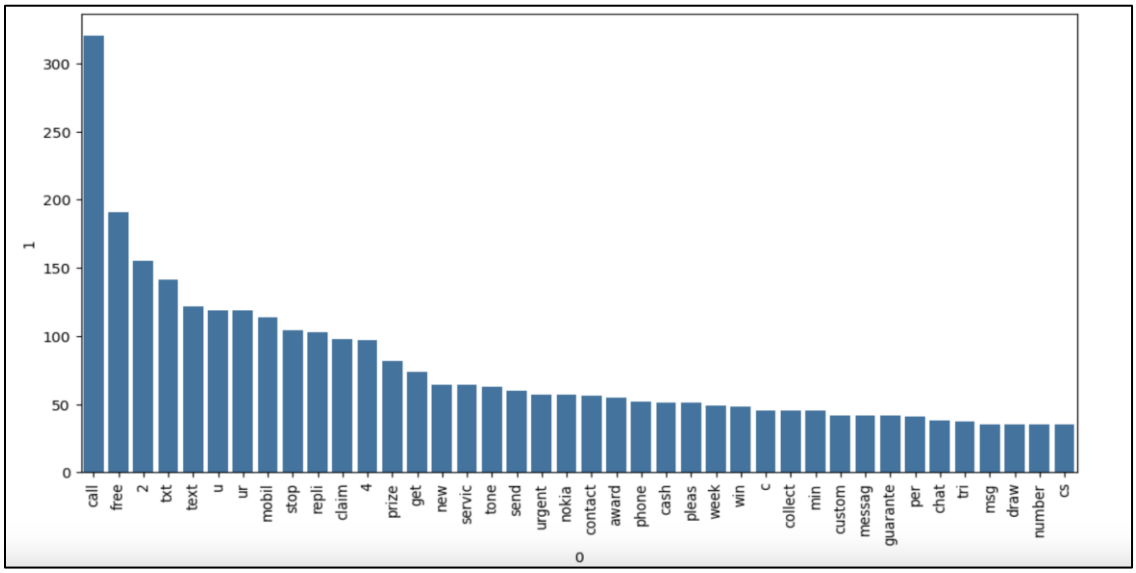


Figure-6: Ham message detection. Data from dataset.

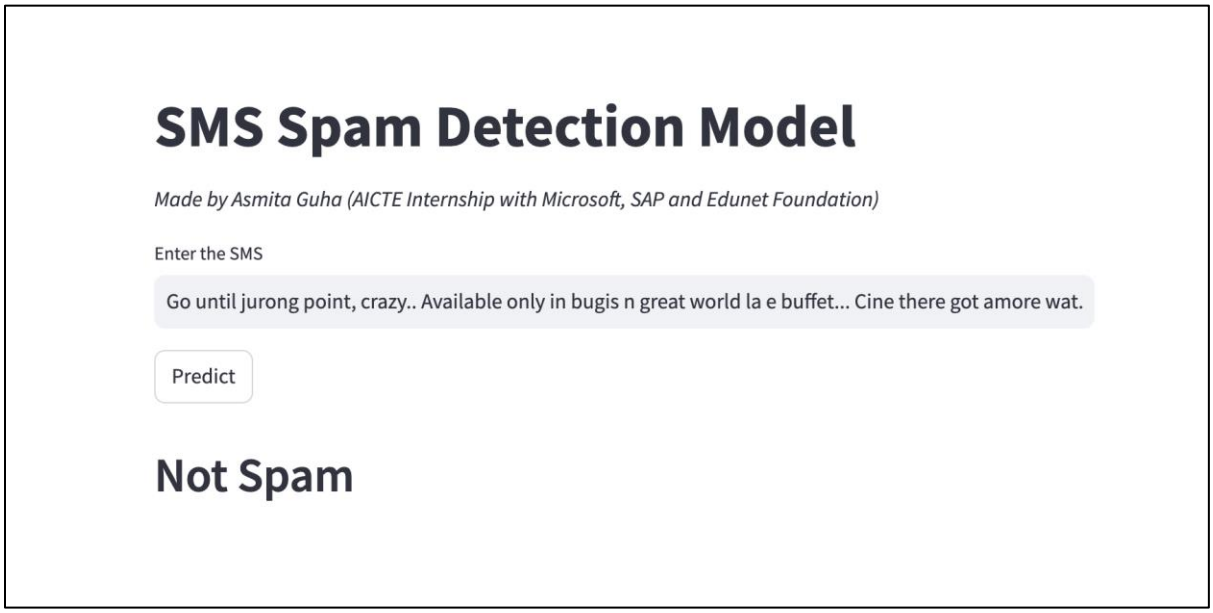


Figure-7: Spam message detection. Data from dataset.

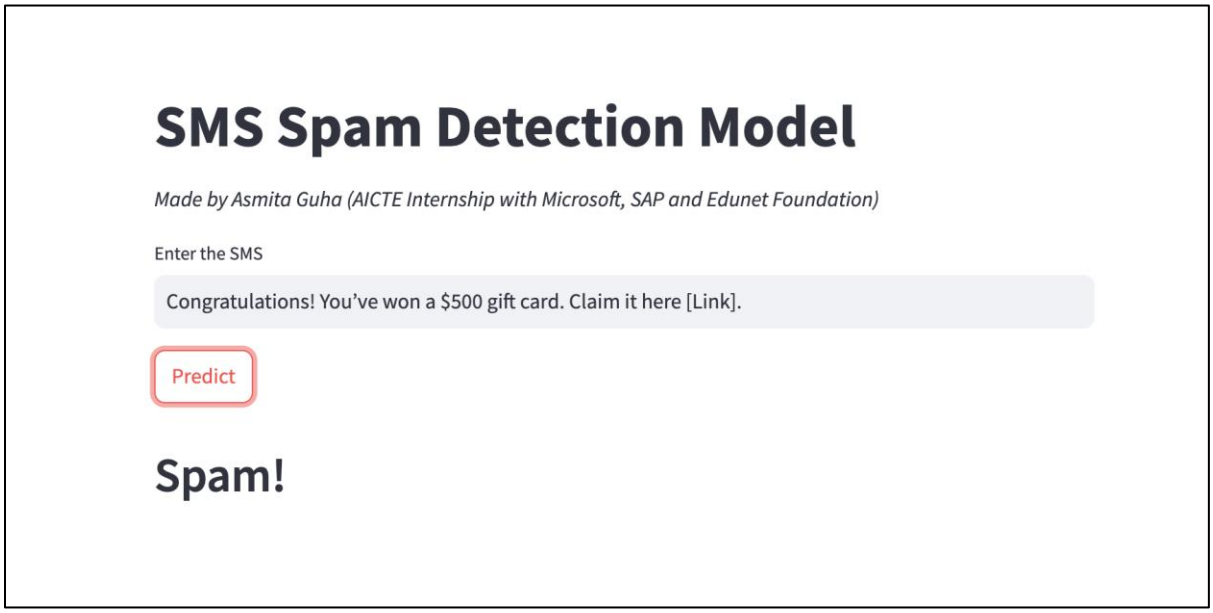


Figure-8: Ham message detection. Data personally provided.

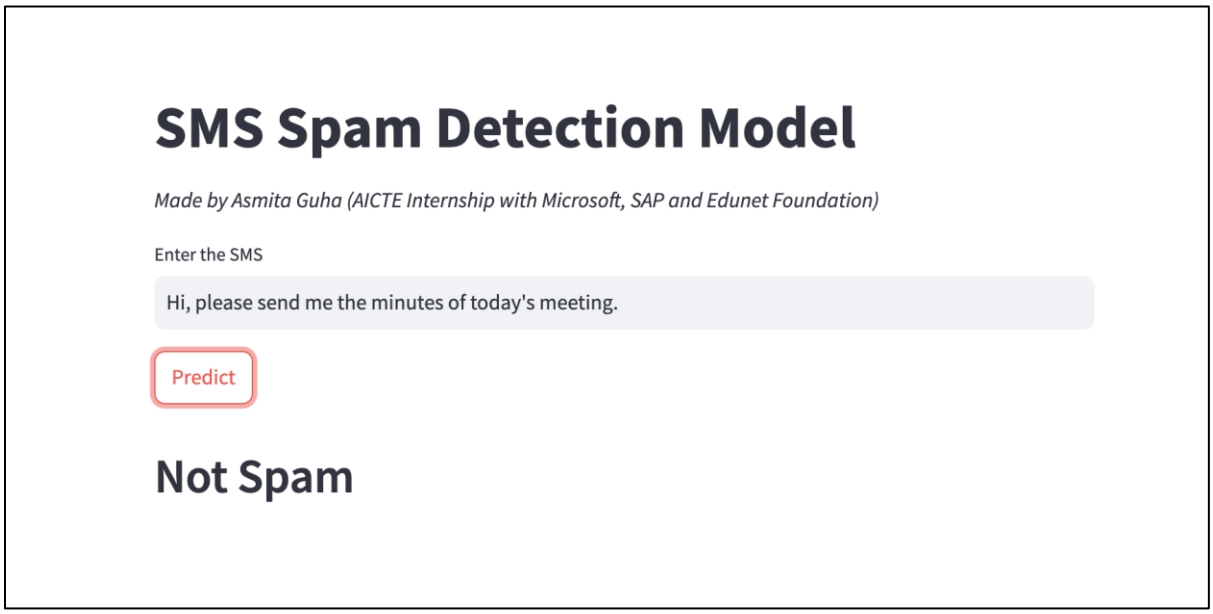
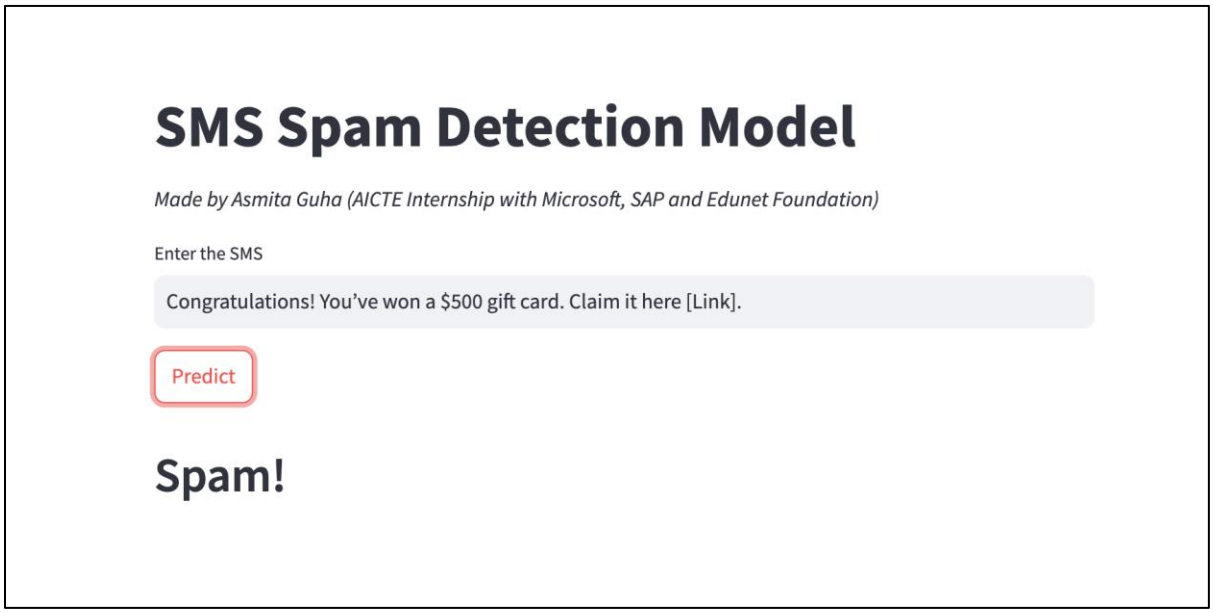


Figure-9: Spam message detection. Data personally provided.



4.2 GitHub Link of the Project:

https://github.com/minnieG12/SMS-Spam-Detection-System

CHAPTER 5

Discussion and Conclusion

5.1 Future Work:

The SMS Spam Detection System has laid a robust foundation for addressing the issue of unsolicited messages. However, there are numerous areas for further improvement and expansion:

1. **Multilingual Dataset Integration:**
   Current systems often focus on English-only datasets, limiting their global applicability. Future work should incorporate multilingual datasets, enabling the system to detect spam messages in diverse languages and regional contexts.
2. **Advanced Spam Tactic Adaptation:**
   a. Spammers continuously evolve their tactics to evade detection. Implementing active learning techniques, where the model adapts based on new data, can help address this challenge.
   b. Adversarial learning can be employed to improve model robustness against sophisticated spam tactics.
3. **Incorporating Transformer Models:**
   a. Models like BERT, GPT-3, or T5 can significantly enhance the contextual understanding of spam messages, offering improved detection accuracy.
   b. Transfer learning techniques using pre-trained Transformer models can be explored to reduce the computational cost of training.
4. **Real-Time Updates and Scalability:**
   a. Automating the data pipeline to include continuous updates with newly labeled data can enhance the system's adaptability.
   b. Scaling the web application to handle a high volume of real-time requests is another area for future improvement.

5. **Mobile and Cloud Deployment:**
    a. Developing mobile applications for Android and iOS platforms can provide users with on-the-go spam detection.
    b. Cloud integration with platforms like AWS, Google Cloud, or Microsoft Azure can enable scalable and efficient deployment.

5.2 Conclusion:

This system of SMS Spam Detection is arguably one of the most promising uses of machine learning and natural language processing in solving social interaction problems as it combines multiple disciplines in one. With high accuracy rates in separating spam versus legitimate messages, the project is poised to showcase the advantages of AI in making users safe and bettering communication.

The deployment of the system as a web application ensures its usefulness in practice as users can enjoy classification of messages in real time. The combination of advanced deep learning models like LSTM with older methods such as Naive Bayes offers a well-rounded as well as powerful spam detection system.

Although this implementation has proven to be effective, the project's flexibility and expansion potential addresses the issue of emerging spam. Further advancement in the areas of integration, architecture, and language support will only make this more effective.

As a result, this project displays the remarkable ability of AI-centric solutions to address problems available today such as SMS spam. There is still a lot of improvement that can be made on the current model, such as making communication safer and more efficient, but the foundation is already there.

REFERENCES

[1] Al-Ghamdi et al., “Spam Email Classification Using ML Techniques,” IJACSA, 2020.
[2] Bird et al., “Natural Language Processing with Python,” O’Reilly Media, 2009.
[3] Streamlit Documentation, “Build Data Apps in Python.”
[4] Dhamija et al., “A Review of SMS Spam Detection Techniques,” IEEE Transactions, 2017.
[5] Salim et al., “Deep Learning for Spam Classification,” Journal of AI Research, 2018.

THANK YOU

Asmita Guha