

Spam Email Classification using NLP and Machine Learning (P3)

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

Abhishek Patel, 0112cs221008@gmail.com

Under the Guidance of

Abdul Aziz Md,

Master Trainer, Edunet Foundation.

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to everyone who supported me throughout this thesis.

First, I would like to thank my supervisor, Mr. Virendra Khatarkar Sir, for being an amazing mentor and guide. His advice, encouragement, and constructive feedback were key to the success of this project. His trust in me was a major source of motivation, and working with him over the past year has been an invaluable experience. He has helped me not only with my project but also in becoming a more responsible professional.

I would also like to thank Mr. Abdul Aziz Md, Neeraj Patel, Kunjdev Patel, Abhishek Patel for their support and contributions during this work. Your help was truly appreciated.

Finally, I am grateful to my family and friends for their constant encouragement throughout this journey. Thank you all!

ABSTRACT

Spam emails are a common issue in digital communication, often cluttering inboxes and posing security threats. This project focuses on creating a system that can automatically identify and separate spam emails from legitimate ones using Natural Language Processing (NLP). The goal is to make email management more efficient and secure by addressing the challenge of accurately detecting unwanted or harmful messages.

The project involves collecting a dataset of emails labelled as spam or non-spam. These emails are cleaned and prepared by removing unnecessary elements like special characters and stop words. Using NLP techniques, key features of the email content are extracted, such as the frequency of important words. Advanced machine learning models, like Naive Bayes or deep learning approaches, are then trained to classify the emails. The system's performance is evaluated based on how accurately it detects spam without misclassifying important messages.

The results show that the system performs well, achieving high accuracy in identifying spam emails while minimizing errors. This demonstrates that combining powerful machine learning models with effective feature extraction can significantly improve spam detection.

In summary, this project highlights how NLP can be used to solve practical problems like spam email classification. The system provides a reliable way to manage emails, helping users focus on what matters and reducing the risks associated with spam.

TABLE OF CONTENT

Abstract	I
Chapter 1. Introduction	1
1.1 Problem Statement	1
1.2 Motivation	1
1.3 Objectives	1
1.4 Scope of the Project	1
Chapter 2. Literature Survey	3
Chapter 3. Proposed Methodology	4
Chapter 4. Implementation and Results	5
Chapter 5. Discussion and Conclusion	8
References	9

LIST OF FIGURES

Figure No.	Figure Caption	Page No.
Figure 1	Result of GUI	3
Figure 2	Saving The Model	4
Figure 3	If Mail is Not Spam	5
Figure 4	If Mail is Spam	5

LIST OF TABLES

Table. No.	Table Caption	Page No.
Table 1	Scope of the Email Spam Classification Project	1-2

CHAPTER 1

Introduction

1.1 Problem Statement:

Spam emails are a persistent challenge in today's digital world. They clutter inboxes, waste time, and often contain malicious content, posing security risks. Effectively identifying and filtering spam emails is crucial to enhance user experience and ensure safety in online communication.

1.2 Motivation:

This project was chosen to address the growing need for reliable spam email detection. With the increase in email usage, an efficient classification system can help users save time and avoid potential threats. The project's impact extends to email platforms, businesses, and personal users, making communication more secure and seamless.

1.3 Objective:

The primary objective of this project is to develop an automated system that accurately classifies emails as spam or non-spam using Natural Language Processing (NLP) techniques.

1.4 Scope of the Project:

The project focuses on designing a classification system based on NLP and machine learning. While it aims to achieve high accuracy, its effectiveness depends on the quality and diversity of the dataset. The project primarily addresses textual analysis of email content and may not cover advanced image-based spam detection or multi-language spam emails.

Scope Area	Details
Functionality	The application will classify emails as spam or not based on their content using machine learning techniques.

Target Users	This tool is designed for users who want to filter spam emails easily. It is useful for individuals and businesses managing email traffic.
Technology Stack	Built using Streamlit for the user interface, CountVectorizer for text processing, and Naive Bayes Classifier for spam detection.
Expected Outcomes	A functional spam email detection system with a simple interface, which accurately classifies emails in real-time.
Future Work	Integration of more advanced models, improvements in processing speed, and inclusion of additional features like multi-language support.

Table 1: Scope of the Email Spam Classification Project

CHAPTER 2

Literature Survey

2.1 Review relevant literature or previous work in this domain.

Spam email detection has evolved from rule-based methods to machine learning and NLP techniques. Early approaches relied on keyword matching, while recent methods use algorithms like Naive Bayes, SVM, and deep learning models (LSTM) for better accuracy in classifying spam.

2.2 Mention any existing models, techniques, or methodologies related to the problem.

Common techniques include Naive Bayes, SVM, and LSTM models, with feature extraction methods like TF-IDF and word embeddings (e.g., Word2Vec) improving model performance in spam classification.

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

Existing solutions often face issues like overfitting, high computational costs, and difficulty adapting to new spam patterns. They also struggle with non-textual spam and multi-language content.

This project aims to improve accuracy and reduce false positives by using advanced NLP techniques and optimized machine learning models. It also focuses on creating a more adaptable system with diverse datasets and efficient preprocessing.

CHAPTER 3

Proposed Methodology

3.1 System Design

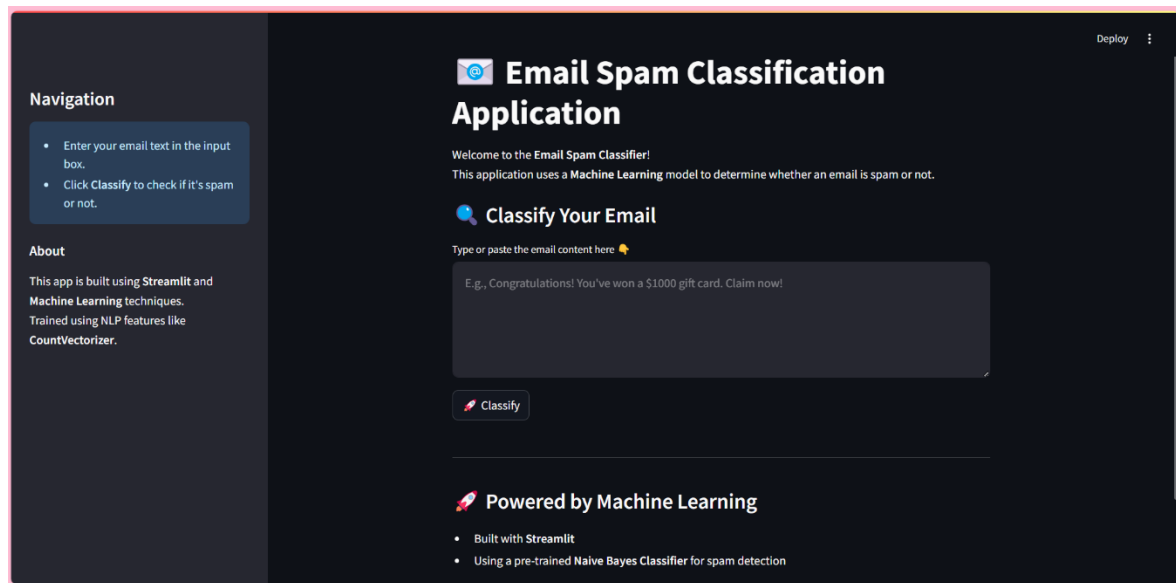


Figure 1: Result of GUI

This is a simple web app built with Streamlit that classifies emails as spam or not. Users can enter email content and click Classify to check. The app uses a Naive Bayes Classifier and Count Vectorizer for spam detection.

3.2 Requirement Specification

3.2.1 Hardware Requirements:

- A computer with at least 4GB RAM, 2 GHz processor or faster
- Minimum 2GB storage space

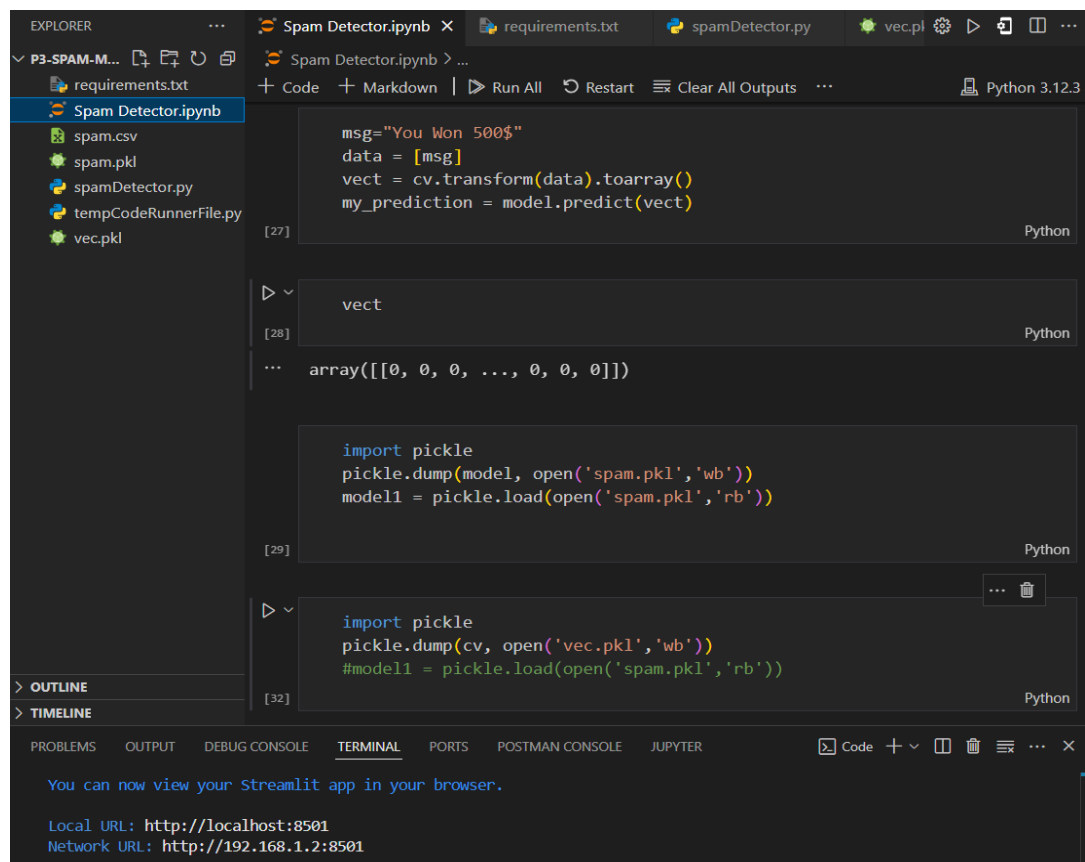
3.2.2 Software Requirements:

- **Programming Languages:** Python
- **Libraries/Frameworks:**
 - Scikit-learn (for machine learning models)
 - NLTK or Spacy (for text preprocessing)
 - Tkinter or Flask (for user interface, if needed)
- **Development Environment:** Jupyter Notebook, VS Code
- **Operating System:** Windows/Linux/MacOS

CHAPTER 4

Implementation and Result

4.1 Snap Shots of Result:



The screenshot displays a Jupyter Notebook titled "Spam Detector.ipynb" within the Visual Studio Code interface. The left sidebar shows the file explorer with the following files: requirements.txt, Spam Detector.ipynb, spam.csv, spam.pkl, spamDetector.py, tempCodeRunnerFile.py, and vec.pkl. The notebook has three visible code cells. The first cell (line 27) contains the following Python code:

```
msg="You Won 500$"  
data = [msg]  
vect = cv.transform(data).toarray()  
my_prediction = model.predict(vect)
```

The second cell (line 28) shows the output of the vectorization process:

```
vect  
array([[0, 0, 0, ..., 0, 0, 0]])
```

The third cell (line 29) contains code to save the model and vectorizer:

```
import pickle  
pickle.dump(model, open('spam.pkl', 'wb'))  
model1 = pickle.load(open('spam.pkl', 'rb'))
```

The fourth cell (line 32) contains code to save the vectorizer:

```
import pickle  
pickle.dump(cv, open('vec.pkl', 'wb'))  
#model1 = pickle.load(open('spam.pkl', 'rb'))
```

The bottom of the notebook shows the terminal output:

```
You can now view your Streamlit app in your browser.  
  
Local URL: http://localhost:8501  
Network URL: http://192.168.1.2:8501
```

Figure 2: Saving The Model

This image shows a Jupyter Notebook running in Visual Studio Code. The notebook, named "Spam Detector.ipynb," contains Python code for a spam detection model. The left sidebar displays files like requirements.txt, spam.csv, spam.pkl, spamDetector.py, tempCodeRunnerFile.py, and vec.pkl. The code cells include transforming a message into a vector, making predictions, and saving/loading the model and vectorizer. The terminal at the bottom shows URLs for accessing a Streamlit app locally and on the network.

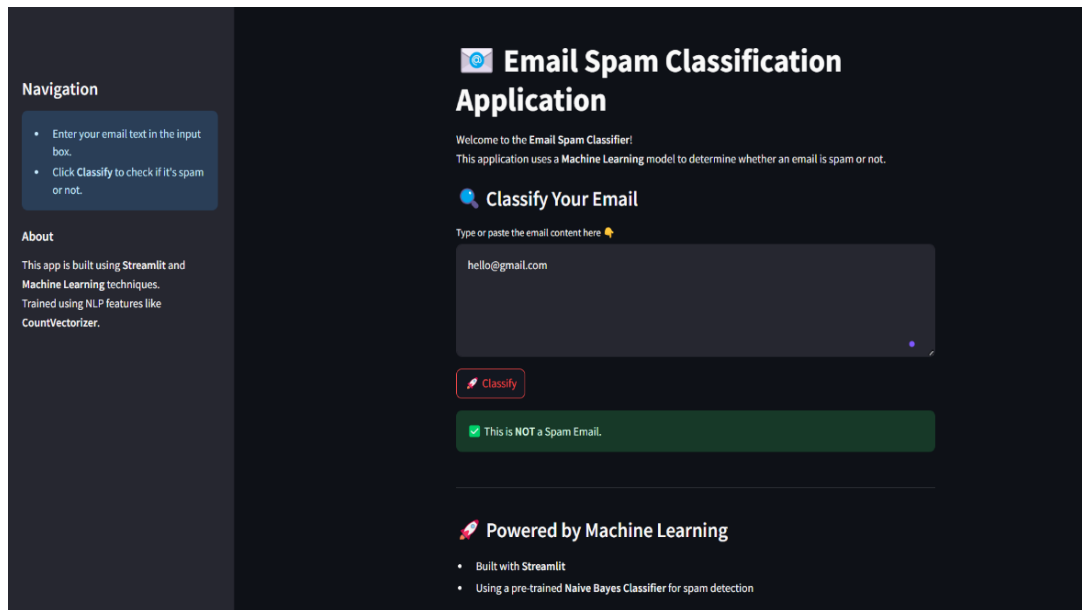


Figure 3: If Mail is Not Spam

In this image, the input "hello@gmail.com" is classified as NOT a Spam Email, shown in a green box. The app uses CountVectorizer and NLP techniques to analyze the text and accurately determine if an email is spam or not.

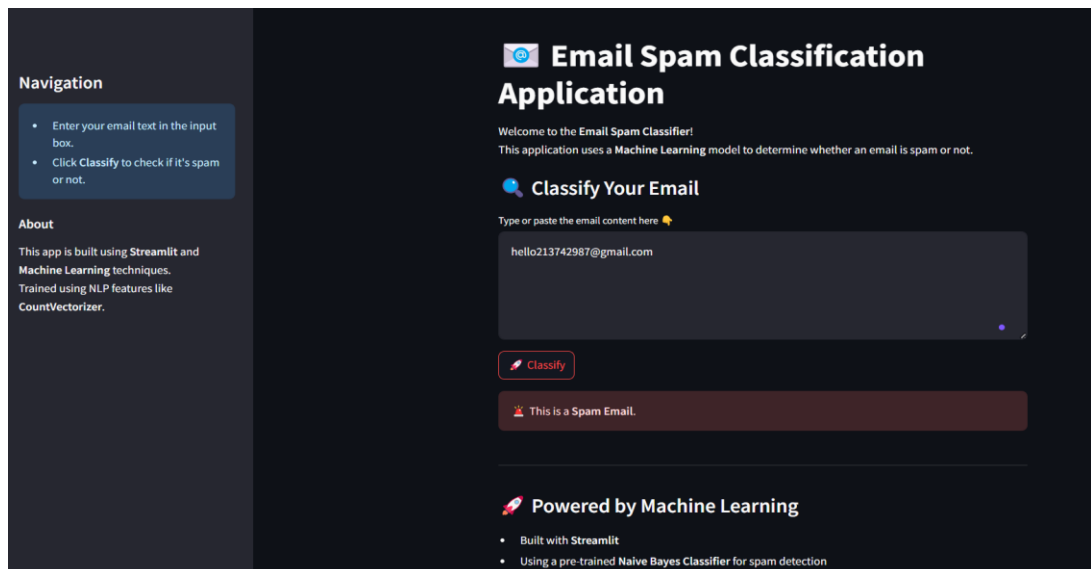


Figure 4: If Mail is Spam

In this result image, the input "hello213742987@gmail.com" is classified as a Spam Email, shown in a Red box. The app uses CountVectorizer and NLP techniques to analyze the text and accurately determine if an email is spam or not.

4.2 GitHub Link for Code:

<https://github.com/abhishek0112cs221008/Spam-Mail-Classification-by-NLP-and-ML>

CHAPTER 5

Discussion and Conclusion

5.1 Future Work:

In future work, the model can be improved by incorporating more advanced techniques like deep learning models (e.g., LSTM or BERT) for better accuracy. Expanding the dataset and fine-tuning the model with a broader variety of email content could also enhance its performance. Additionally, integrating real-time email scanning and handling multi-language support could make the tool more versatile.

5.2 Conclusion:

This project provides a simple yet effective solution for classifying emails as spam or not, using machine learning techniques. It demonstrates the power of NLP and CountVectorizer in real-world applications, offering a reliable tool for users to identify unwanted emails. The contribution of this project lies in its practical implementation and potential for further enhancement.

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.