**Spam Email Classification Using NLP and Machine Learning**

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

**Shashank Mishra**

**Mishrasam012@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 during my internship.

First, I want to thank my supervisor, Abdul Aziz Md, for his guidance and support throughout this journey. His advice and encouragement have been invaluable, helping me learn and grow both professionally and personally. Working under his mentorship has been a truly rewarding experience.

I am also grateful to the Edunet Foundation for providing me with this internship opportunity. Their platform and resources have played a key role in enhancing my skills and knowledge.

A special thanks to my friend, Nirnay Awasthi, for his assistance in helping me secure this internship. His support and belief in my abilities made this opportunity possible, and I am truly thankful for his help.

This internship has been a great learning experience, and I deeply appreciate the guidance and support from Abdul Aziz Md, the Edunet Foundation, and my friend Nirnay Awasthi.

#### **ABSTRACT**

Nowadays communication plays a key role in everything be it professional or personal. Email communication service is being used extensively because of its free use services, low-cost operations, accessibility, and popularity. Emails have one major security flaw that is anyone can send an email to anyone just by getting their unique user id. This security flaw is being exploited by some businesses and ill-motivated persons for advertising, phishing, malicious purposes, and finally fraud. This produces a kind of email category called SPAM.

Spam refers to any email that contains an advertisement, unrelated and frequent emails. These emails are increasing day by day in numbers. Studies show that around 55 percent of all emails are some kind of spam. A lot of effort is being put into this by service providers. Spam is evolving by changing the obvious markers of detection. Moreover, the spam detection of service providers can never be aggressive with classification because it may cause potential information loss in case of a misclassification.

To tackle this problem, we present a new and efficient method to detect spam using machine learning and natural language processing. A tool that can detect and classify spam. In addition to that, it also provides information regarding the text provided in a quick view format for user convenience.

**TABLE OF CONTENT**

**Abstract** **I**

**Chapter 1.**  **Introduction 1**

1.1 Problem Statement 1

1.2 Motivation 1

1.3 Objectives 2

1.4. Scope of the Project 2

1.5 Limitation of the Project 2

**Chapter 2.**  **Literature Survey** **3**

2.1 Introduction 3

2.2 Related Work 3

**Chapter 3.**  **Proposed Methodology** **4**

3.1 System Design/Architecture 4

3.1.1 Modules and Architecture Design 4-5

3.2 Requirement Specifications 6

3.2.1 Hardware Requirements 6

3.2.2 Software Requirements 6

3.3 Models and Algorithms 6

3.3.1 Naïve Bayes Classifier 6-7

3.3.2 Multinomial Naive Bayes (NB) 7

3.3.3 Scikit-learn 7

**Chapter 4.**  **Implementation and Results**  **8**

4.1 Snap Shots of Result 8

4.1.1 Email Spam Classification Application 8-9

4.1.2 How to use Email Spam Classification application 9-10

4.2 GitHub Link for Code and Application 10

**Chapter 5.**  **Discussion and Conclusion**  **11**

5.1 Future Work 11

5.2 Conclusion 11

5.3 Overall Impact and Contribution 12

**References** **13**

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Figure Caption** | **Page No.** |
| **Figure-3.1** | **System Architecture** | **4** |
| **Figure-4.1** | **Email Spam Classification Application (Spam Mail)** | **8** |
| **Figure-4.2** | **Email Spam Classification Application (Ham Mail)** | **9** |

**CHAPTER 1**

**Introduction**

* 1. **Problem Statement:**

Email spam remains a persistent challenge as spammers continuously evolve their tactics to bypass filtering systems, using obfuscation, and embedding tricks. Traditional spam filters struggle to balance accuracy, often leading to either false positives that block important emails or false negatives that allow spam to reach users. This problem demands a more robust and precise classification approach to effectively identify and mitigate spam while preserving critical communication.

* 1. **Motivation:**

The rapid increase in the volume of emails in daily communication has brought attention to the growing problem of **spam emails**. Studies indicate that **around 55% of all emails are spam**, and this number continues to rise, with spammers evolving their techniques to bypass traditional detection systems. Spam emails can lead to several consequences, including wasted time, security breaches, phishing attempts, and financial fraud. Furthermore, service providers must ensure that spam detection mechanisms are accurate enough not to misclassify legitimate communications as spam, as this could lead to a loss of critical information for users.

Machine Learning (ML) and Natural Language Processing (NLP) offer promising solutions to this challenge by enabling automated spam detection through efficient analysis of email patterns, language use, and trends. However, the constant adaptation of spam tactics makes it imperative to use advanced, adaptive, and ensemble-based machine learning models that combine multiple techniques to ensure higher accuracy rates.

This project was chosen to address the limitations of conventional spam detection methods by leveraging **machine learning models with NLP techniques** to classify spam emails efficiently. The primary motivation is to create a system that detects spam with minimal false positives while maximizing detection accuracy. This project has the potential to save users time, enhance email security, and improve the reliability of communication systems by providing a scalable and intelligent spam filtering mechanism.

The outcomes of this research can extend to various real-world applications such as automated email filtering for enterprises, reducing phishing vulnerabilities, and even analyzing patterns in public communication data.

* 1. **Objective:**

Clearly state the objectives of the project.

1. To create an ensemble algorithm for classification of spam with highest possible accuracy.
2. To study how to use machine learning for spam detection.
3. To study how natural language processing techniques can be implemented in spam detection.
4. To provide user with insights of the given text leveraging the created algorithm and NLP.
   1. **Scope of the Project:**

This project needs a coordinated scope of work.

1. Combine existing machine learning algorithms to form a better ensemble algorithm.
2. Cleaning, processing and making use of the dataset for training and testing the model created.
3. Analyze the texts and extract entities for presentation.
   1. **Limitation of the Project:**

This Project has certain limitations.

1. Analysis can be tricky for some alphanumeric messages, and it may struggle with entity detection.
2. Since the data is reasonably large it may take a few seconds to classify and analyze the message.
3. Since the data is reasonably large it may take a few seconds to classify and analyze the message.

**CHAPTER 2**

**Literature Survey**

* 1. **Introduction**

This chapter discusses the literature review for machine learning classifier that being used in previous research and projects. It is not about information gathering but it summarizes the prior research that related to this project. It involves the process of searching, reading, analyzing, summarizing and evaluating the reading materials based on the project.

A lot of research has been done on spam detection using machine learning. But due to the evolvement of spam and development of various technologies the proposed methods are not dependable. Natural language processing is one of the lesser-known fields in machine learning and it is reflected here with comparatively less work present.

* 1. **Related Work**

Spam classification is a problem that is neither new nor simple. A lot of research has been done, and several effective methods have been proposed.

1. M. RAZA, N. D. Jayasinghe, and M. M. A. Muslam have analyzed various techniques for spam classification and concluded that naïve Bayes and support vector machines have higher accuracy than the rest, around 91% consistently.
2. S. Gadde, A. Lakshmanarao, and S. Satyanarayana in their paper on spam detection concluded that the LSTM system resulted in higher accuracy of 98%.
3. P. Navanya, G. Dubey, and A. Rana compared the efficiency of the SVM, naïve Bayes, and entropy method and the SVM had the highest accuracy (97.5%) compared to the other two models.

From numerous studies, we can see that for several types of data various models perform better. Naïve Bayes, random forest, SVM, logistic regression is some of the most used algorithms in spam detection and classification.

**CHAPTER 3**

**Proposed Methodology**

* 1. **System Design/Architecture**

The overview has been presented below, and it gives a basic architecture of the application.

Fig 1 System Architecture
 Fig.no. 3.1- System Architecture

**3.1.1 Modules and Architecture Explanation**

Here is a breakdown of each step in the flowchart:

### Input:

1. **Receive Email:**
   1. The process begins with receiving an email to be analyzed.
2. **Preprocess Email:**
   1. Cleans and normalizes the email content.
   2. Removes unnecessary characters, formatting, or irrelevant information (e.g., HTML tags).

### NLP (Natural Language Processing):

1. **Tokenize Text:**
   1. Splits the email text into smaller units called tokens (e.g., words or phrases).
   2. Helps to analyze the text content more effectively.
2. **Remove Stop Words:**
   1. Eliminates frequently used words like "the," "is," and "and" that do not contribute to the context.
   2. Reduces noise in the data for better analysis.
3. **Stemming:**
   1. Reduces words to their root form (e.g., "running" becomes "run").
   2. Ensures similar words are grouped together to improve consistency.

### Feature Extraction:

1. **Vectorize Text:**
   1. Converts the processed text into a numerical representation (e.g., Bag of Words or TF-IDF).
   2. Prepares the data for machine learning.
2. **Extract Features:**
   1. Identifies and selects key features (e.g., frequency of certain words or patterns) from the vectorized text.
   2. These features are used by the ML model to make predictions

### ML Model:

1. **Load Trained Model:**
   1. Loads a pre-trained machine learning model designed to classify emails as spam or not spam.
   2. The model is trained on labeled datasets of emails.
2. **Predict Spam or Not:**
   1. Based on the extracted features, the model predicts whether the email is spam or not.

### Output:

1. **Mark as Spam:**
   1. If the prediction classifies the email as spam, it is flagged as such.
2. **Mark as Not Spam:**
   1. If the prediction identifies the email as legitimate, it is marked as not spam.

This entire pipeline ensures that emails are accurately classified for effective spam filtering.

* 1. **Requirement Specification**

Mention the tools and technologies required to implement the solution.

* + 1. **Hardware Requirements:**
* PC/Laptop
* Ram – 8 GB
* Storage – 100-200 MB
  + 1. **Software Requirements:**
* OS – Windows 7 and above
* Code Editor – PyCharm, VS Code, Built in IDE
* Anaconda environment with packages Streamlit, Pickle, NumPy, pandas, Sklearn, skinter, nltk data etc.
* Supported browser such as Chrome, Firefox, Opera, Brave etc.
  1. **Models and Algorithms**

Model and Algorithms used for the classifications.

**3.3.1 Naïve Bayes Classifier**

A naïve Bayes classifier is a supervised probabilistic machine learning model that is used for classification tasks. The main principle behind this model is the Bayes theorem.

**Bayes Theorem:**

Naive Bayes is a classification technique that is based on Bayes’ Theorem with an assumption that all the features that predict the target value are independent of each other. It calculates the probability of each class and then picks the one with the highest probability.

Naive Bayes classifier assumes that the features we use to predict

the target is independent and do not affect each other.

Though the independence assumption is never correct in real-world data but often works well in practice. so that it is called “Naive.”

**P(A│B) =(P(B│A) P(A))/P(B)**

P(A|B) is the probability of hypothesis A given the data B. This is called the posterior probability.

P(B|A) is the probability of data B given that hypothesis A was true.

P(A) is the probability of hypothesis A being true (regardless of the data). This is called the prior probability of A.

P(B) is the probability of the data (regardless of the hypothesis).

Naïve Bayes classifiers are mostly used for text classification. The limitation of the Naïve Bayes model is that it treats every word in a text as independent and is equal in importance, but every word cannot be treated as equally important because articles and nouns are not the same when it comes to language. But due to its classification efficiency, this model is used in combination with other language processing techniques.

**3.3.2 Multinomial Naive Bayes (NB):**

The Multinomial Naive Bayes (NB) classifier is a variant of the Naive Bayes algorithm specifically designed for discrete data, such as word counts in text classification tasks. It assumes that features (e.g., words) follow a multinomial distribution, where the frequency of each word contributes to the probability of a class. The classifier calculates the posterior probability of a class using Bayes' theorem, based on the prior probability and likelihood of features. Multinomial NB is particularly effective for text-based applications like spam detection, sentiment analysis, and document classification, where feature frequencies (e.g., term frequencies) play a critical role.

**3.3.3 Scikit-learn**

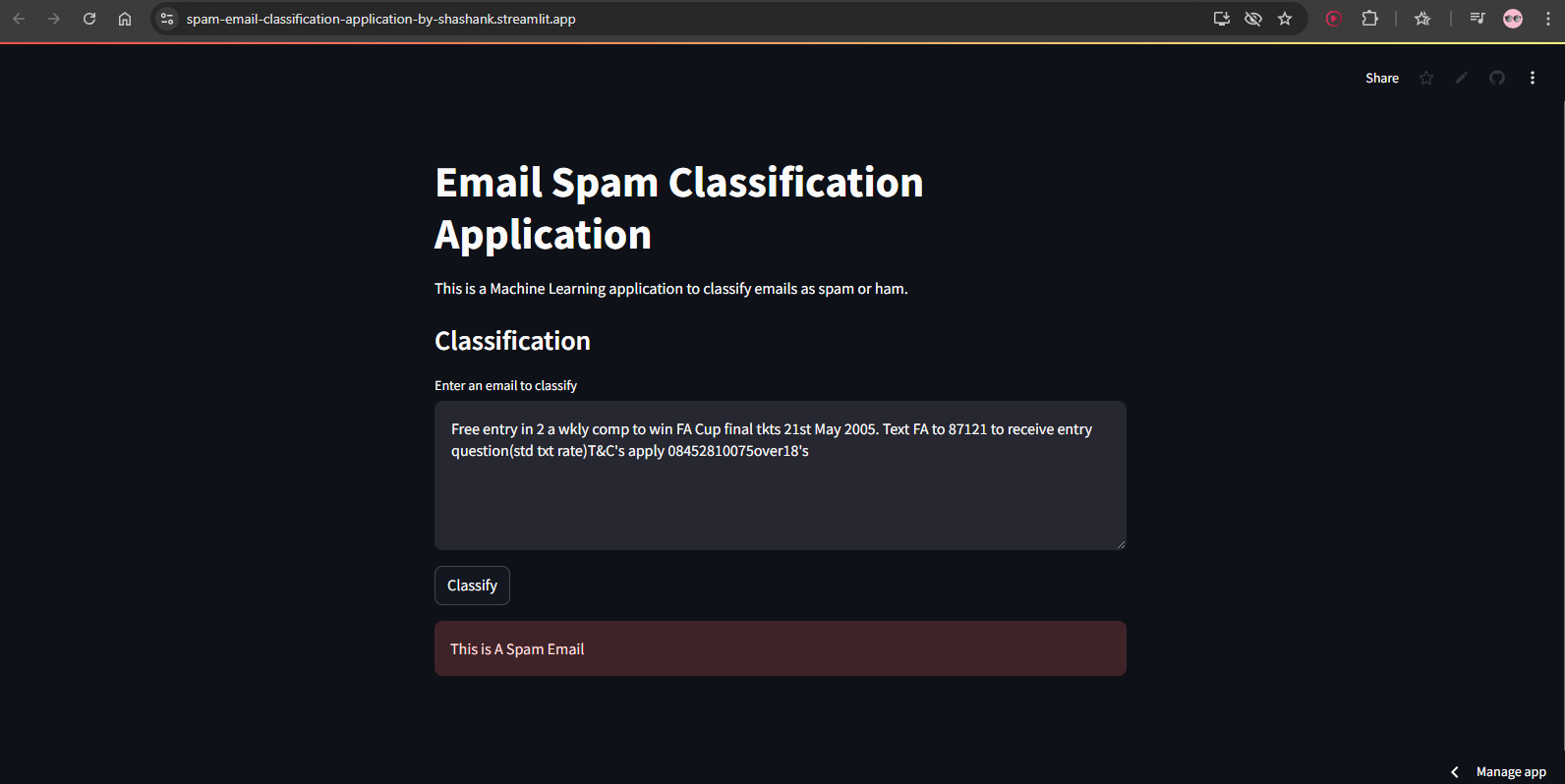
“Sklearn.naive\_bayes” is a module in the **scikit-learn** library that implements different types of Naive Bayes classifiers for supervised machine learning tasks, especially for classification problems involving text data (e.g., spam detection, sentiment analysis). It provides efficient tools for implementing **Naive Bayes** models in Python.

**CHAPTER 4**

**Implementation and Result**

* 1. **Snap Shots of Result:**

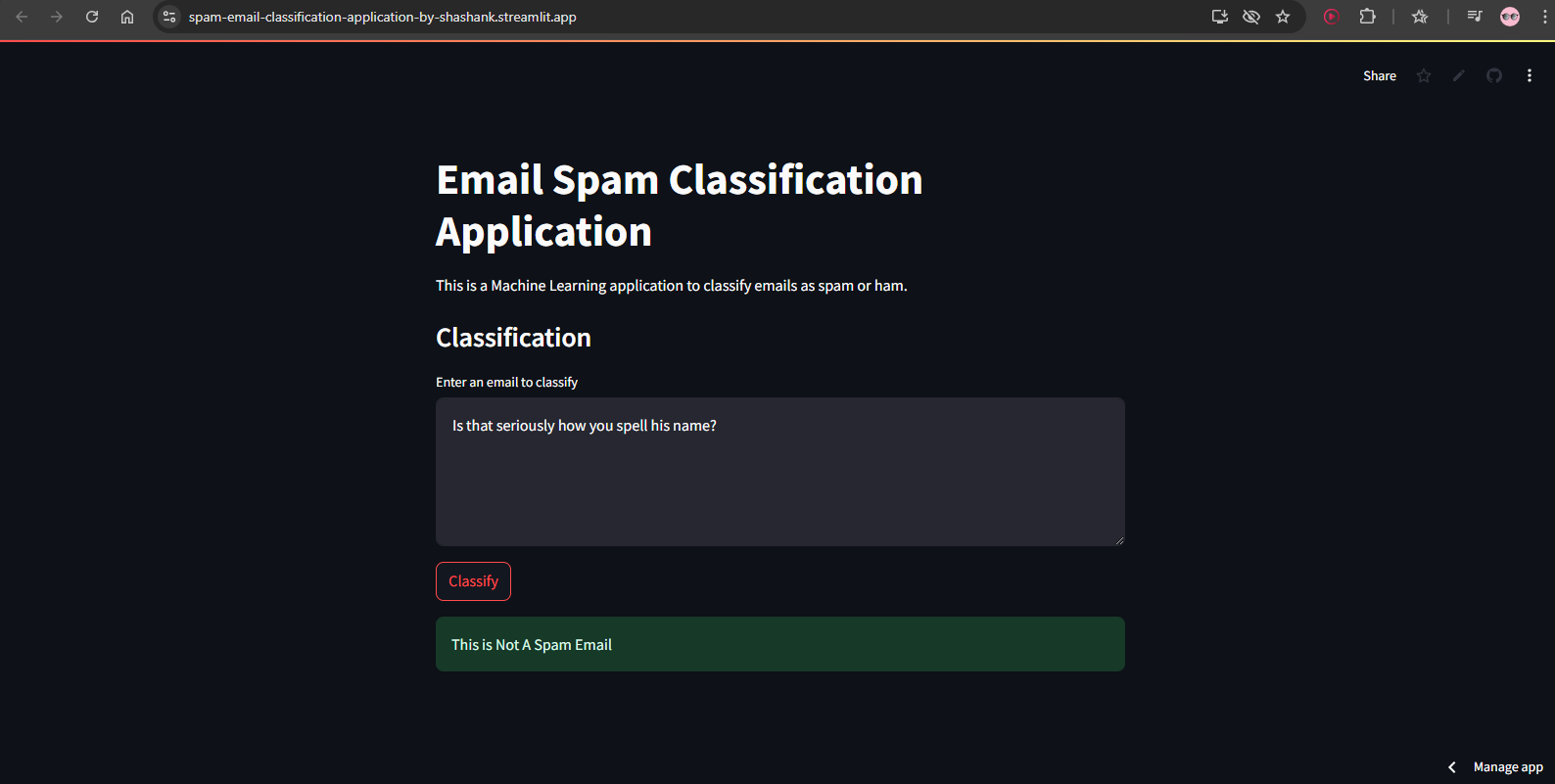
**4.1.1 Email Spam Classification Application**

Fig.no.4.1- Email Spam Classification Application (Spam Mail)

Email Spam Classification Application designed to identify whether a given email is spam or legitimate (ham). Built using a Machine Learning model, it provides a user-friendly interface for text classification tasks. Users input the content of an email in the provided text box, and upon clicking the "Classify" button, the system analyzes the text and displays the result.

In this example, the application processes a sample email offering a prize, which is identified as spam. The model likely uses natural language processing techniques and algorithms like Naive Bayes or SVM, trained on labeled datasets, to classify the content accurately.

The application is hosted on Streamlit, a popular framework for deploying machine learning models and interactive applications. Its minimalistic design and intuitive workflow make it suitable for testing ML model performance, highlighting practical applications of machine learning, or serving as a simple tool for non-technical users to identify spam emails. The clean, dark-themed layout enhances usability and readability, offering a professional platform for spam detection tasks.

Fig.no.4.1- Email Spam Classification Application (Ham Mail)

This dashboard features an Email Spam Classification Application designed to classify emails as either spam or ham (non-spam). Users input the content of an email into the text box, and the system processes it to determine its classification. For example, the entered text, is identified as **not spam**, shown with a green label.

The application leverages a trained Machine Learning model, using natural language processing techniques to distinguish genuine communication from spam messages. Hosted on Streamlit, the tool is designed with a clean, professional interface, making it simple and efficient for users to test email classification. It demonstrates practical applications of machine learning in text analysis, particularly in enhancing email filtering systems.

**4.1.2 How to use Email Spam Classification Application:**

To use this Email Spam Classification Application as a frontend user, follow these steps:

### 1. Access the Application:

* Open the application in your web browser via the provided link or URL (hosted on Streamlit).

### 2. Input Email Content:

* Locate the input box on the interface.
* Enter the email text you want to classify into the text box. You can paste the content or type it manually.

### 3. Trigger Classification:

* Click the **"Classify"** button to process the entered email.

### 4. View the Results:

* The application will analyze the text and display the result:
  + If the email is **spam**, a red box with the text *"This is a Spam Email"* will appear.
  + If the email is not spam (ham), a green box with the text *"This is Not a Spam Email"* will be shown.

### 5. Test More Emails:

* Clear the text box or overwrite the previous text with a new email.
* Click **"Classify"** again to test additional emails.

### 6. Use Cases:

* Identify spam emails for personal or professional purposes.
* Test the model's accuracy by inputting various types of email content.

The interface is user-friendly and does not require any technical expertise.

* 1. **GitHub Link for Code:**
* **Code Link:**

[**https://github.com/ShashankMishra9696/Spam-Email-Classification-Application**](https://github.com/ShashankMishra9696/Spam-Email-Classification-Application)

* **Email Spam Classification Application Link:**

[**https://spam-email-classification-application-by-shashank.streamlit.app/**](https://spam-email-classification-application-by-shashank.streamlit.app/)

**CHAPTER 5**

**Discussion and Conclusion**

* 1. **Future Work:**

There are numerous applications to machine learning and natural language processing and when combined they can solve some of the most troubling problems concerned with texts. This application can be scaled to intake text in bulk so that classification can be done more effectively in some public sites.

Other contexts such as negative, phishing, malicious, etc. can be used to train the model to filter things such as public comments in various social sites. This application can be converted to an online type of machine learning system and can be easily updated with latest trends of spam and other mails so that the system can adapt to new types of spam emails and texts.

* 1. **Conclusion:**

In conclusion, this project successfully developed an intelligent email spam detection system using machine learning (ML) and natural language processing (NLP) techniques. The system leverages advanced machine learning algorithms like **Naïve Bayes**, combined with NLP preprocessing methods to analyze and classify emails as spam or ham (non-spam). Through feature extraction and model training, the system can classify email content accurately by identifying patterns and linguistic structures indicative of spam messages.

The developed system integrates user-friendly features by utilizing **Streamlit**, providing users with an intuitive interface to input email text and instantly determine its classification. This eliminates the need for technical expertise while offering accurate insights into whether an email is spam or legitimate. Furthermore, the ensemble method employed, supported by feature extraction and NLP techniques, ensures higher accuracy and adaptability compared to traditional detection methods.

### 5.3 Overall Impact and Contribution

The project's primary contribution lies in its ability to address the persistent problem of email spam through machine learning, enhancing efficiency in identifying spam messages. The system's ability to minimize false positives and false negatives contributes to improved email security, reduced user frustration, and streamlined communication. Moreover, by integrating NLP methods like tokenization, stop-word removal, and stemming, the system enhances text understanding for better performance in spam detection tasks.

The application has potential real-world use cases, ranging from personal communication security to enterprise-level email filtering and phishing detection. It can also extend into public communication data analysis for pattern detection. This study demonstrates how machine learning models can be combined with NLP preprocessing for scalable, accurate, and intelligent spam detection.

Additionally, the system's modular design allows for future improvements, such as integrating new spam detection trends or expanding the scope of classifications. This ensures the model's adaptability to evolving spam tactics and emerging communication threats. Overall, the system represents a valuable contribution to improving email security and the practical application of machine learning and NLP in real-time text classification.

**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. S. Nandhini and J. Marseline K.S., *"Performance Evaluation of Machine Learning Algorithms for Email Spam Detection,"* in *International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, 2020.
3. N. D. Jayasinghe and M. M. A. Muslam, *"A Comprehensive Review on Email Spam Classification using Machine Learning Algorithms,"* in *International Conference on Information Networking (ICOIN)*, 2021.
4. M. Fattahi, *"SpaML: a Bimodal Ensemble Learning Spam Detector based on NLP Techniques,"* in *IEEE 5th International Conference on Cryptography, Security, and Privacy (CSP)*, 2021.
5. M. Madan, *"medium.com,"* [Online].
6. ChatGPT (OpenAI's GPT-4 Model), 2023.
7. H. Deng, *"Towards Data Science,"* [Online].

Available: <https://towardsdatascience.com/random-forest-3a55c3aca46d>