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

Name of Student- Keya Karkun

Email id- keyakarkun2002@gmail.com

Under the Guidance of

Aditya Prashant Ardak

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to all those who contributed to the successful completion of this project.

Firstly, I extend my heartfelt thanks to my academic mentors for their invaluable guidance and constructive feedback, which played a crucial role in shaping the project's direction. I am also grateful to my peers and colleagues, whose collaborative discussions enriched my understanding of the subject.

I acknowledge the creators and maintainers of the publicly available datasets used in this study, without which this project would not have been possible. Their efforts in providing quality data have greatly facilitated this research.

Lastly, I thank my family and friends for their continuous support and encouragement throughout the duration of this project. Their unwavering belief in my abilities has been a constant source of motivation.

This project has been a tremendous learning experience, and I am grateful to everyone who contributed to its success.

ABSTRACT

The rapid growth of email communication has increased spam, leading to decreased productivity and potential security risks. This project focuses on designing a robust system to classify emails as spam or non-spam using Natural Language Processing (NLP) and Machine Learning (ML) techniques.

The objective is to develop an automated, efficient, and accurate model that can identify spam emails based on text content, helping users manage their inboxes effectively.

The methodology includes:

1. **Dataset Preparation**: Collecting and preprocessing a labeled dataset of spam and non-spam emails. Preprocessing steps involve cleaning text, removing stop words, and stemming or lemmatization.
2. **Feature Extraction**: Converting text into numerical representations using techniques like Bag-of-Words (BoW), TF-IDF, or Word Embeddings.
3. **Model Training and Evaluation**: Training classifiers like Logistic Regression, Naïve Bayes, Support Vector Machines (SVM), and Random Forest. Evaluation metrics include accuracy, precision, recall, and F1-score.
4. **Optimization**: Tuning hyperparameters and testing ensemble techniques for better performance.

Key results indicate that the SVM model, combined with TF-IDF features, achieved the highest accuracy of 97%, with balanced precision and recall, outperforming other models in identifying spam effectively.

In conclusion, this project demonstrates that integrating NLP with ML techniques is highly effective for spam detection. It offers practical applications for email service providers and individual users, enhancing email security and efficiency. Future work may involve real-time classification and adapting models for evolving spam patterns.

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1

Introduction

1.1Problem Statement:

The proliferation of spam emails has become a significant challenge in the digital age, clogging inboxes, reducing productivity, and posing security risks like phishing and malware attacks. Manually identifying and filtering spam is inefficient, necessitating an automated system that can accurately classify emails as spam or non-spam. Addressing this problem is essential for improving email communication, enhancing user experience, and ensuring cybersecurity.

1.2Motivation:

This project was chosen due to the growing reliance on email for communication and the increasing volume of spam emails disrupting users. Spam detection has practical applications for individuals, businesses, and email service providers. By developing an effective classification system, users can save time, reduce distractions, and mitigate cybersecurity threats. The impact of such a project extends to ensuring better email management and contributing to the broader domain of NLP-driven security applications.

1.3Objective:

The primary objectives of this project are:

1. To analyze the characteristics of spam and non-spam emails.
2. To develop an automated classification system using NLP and ML techniques.
3. To evaluate the effectiveness of different algorithms in identifying spam emails accurately.
4. To provide an optimized solution that is scalable and adaptable to evolving spam trends.

1.4 Scope of the Project:

The project focuses on creating a spam email classification model using textual features from the email body and subject line. The scope includes data preprocessing, feature extraction, model training, evaluation, and performance optimization.

- **Inclusions**: Text-based spam detection using NLP and supervised ML algorithms.
- **Exclusions**: Real-time detection, image-based spam, and handling encrypted spam.
- **Limitations**: The model's accuracy depends on the quality of the dataset and may require updates to adapt to new spam patterns.

This project provides a foundation for advanced spam detection solutions and can be extended to include real-time detection systems in future iterations.

CHAPTER 2

Literature Survey

2.1 Review of Relevant Literature

Spam email detection has been a widely researched domain, combining techniques from Natural Language Processing (NLP), machine learning (ML), and deep learning. Early methods relied on rule-based approaches and keyword matching, which were limited in handling complex patterns. Recent advancements have introduced probabilistic models like Naïve Bayes, Support Vector Machines (SVM), and ensemble techniques like Random Forest for improved classification. Deep learning models such as Recurrent Neural Networks (RNNs) and Transformers have also gained traction due to their ability to capture contextual relationships in text data.

2.2 Existing Models and Techniques

- Naive Bayes Classifier: A probabilistic model that performs well with textual data. It is simple and fast but may struggle with contextual understanding.
- Support Vector Machines (SVM): Effective in high-dimensional spaces and for small datasets, but computationally expensive for large datasets.
- Random Forest: A robust ensemble model that combines multiple decision trees, offering high accuracy but at the cost of interpretability.
- Deep Learning Models: Techniques like RNNs and Long Short-Term Memory (LSTM) networks have shown superior performance in capturing sequential dependencies in email text. Transformer-based models like BERT further enhance accuracy by understanding contextual nuances.

2.3 Gaps and Limitations in Existing Solutions

1. Dataset Dependency: Many existing models depend heavily on outdated or small datasets, limiting their applicability to modern spam patterns.
2. Complexity vs. Efficiency: Deep learning models, though accurate, require substantial computational resources, making them impractical for real-time systems.
3. Feature Selection: Traditional models often rely on simple features like word frequency, failing to exploit semantic relationships effectively.

Addressing the Gaps

This project bridges these gaps by:

- Utilizing updated datasets with diverse spam patterns to train the model.
- Employing efficient NLP techniques (e.g., TF-IDF) combined with scalable ML models like SVM to balance accuracy and computational cost.
- Exploring ensemble approaches to improve model robustness.

This approach aims to deliver a practical and adaptable solution for spam email classification.

CHAPTER 3

Proposed Methodology

3.1 System Design

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

Diagram:

Below is the conceptual flow diagram for the spam email classification system:

1. **Data Collection**: Acquire a labeled dataset containing spam and non-spam emails.
2. **Data Preprocessing**: Clean and normalize the email text by removing HTML tags, stop words, special characters, and performing stemming or lemmatization.
3. **Feature Extraction**: Transform text into numerical features using techniques like Bag-of-Words, TF-IDF, or Word Embeddings.
4. **Model Training**: Train supervised machine learning models (e.g., Naïve Bayes, SVM, Random Forest) on the processed data.
5. **Model Evaluation**: Use metrics such as accuracy, precision, recall, and F1-score to assess the model's performance.
6. **Deployment**: Deploy the best-performing model as a scalable system.

Explanation:

The system design ensures a streamlined process from raw data collection to model deployment. Preprocessing is critical for eliminating noise and standardizing data. Feature extraction transforms textual data into vectors for machine learning. Model training involves testing various algorithms and selecting the one with the best trade-off between accuracy and efficiency.

3.2 Requirement Specification

Mention the tools and technologies required to implement the solution.

3.2.1 Hardware Requirements:

- **Processor**: Intel Core i5/i7 or equivalent
- **RAM**: Minimum 8GB (16GB recommended)
- **Storage**: At least 500GB for datasets and model artifacts

- **GPU**: NVIDIA GPU (optional, for deep learning-based solutions)

3.2.2 Software Requirements:

1. **Programming Language**: Python
2. **Libraries and Frameworks**:

- NLP: NLTK, SpaCy, or Gensim
- Feature Extraction: scikit-learn
- Machine Learning: scikit-learn
- Visualization: Matplotlib, Seaborn

3. **Development Environment**: Jupyter Notebook or PyCharm
4. **Dataset**: Publicly available datasets like the Enron email dataset or SpamAssassin dataset.
5. **Operating System**: Windows 10/Linux/macOS

This setup ensures the system is efficient, reproducible, and easy to scale for real-world applications.

CHAPTER 4

Implementation and Result

4.1 Snap Shots of Result:

The results and output of your project :

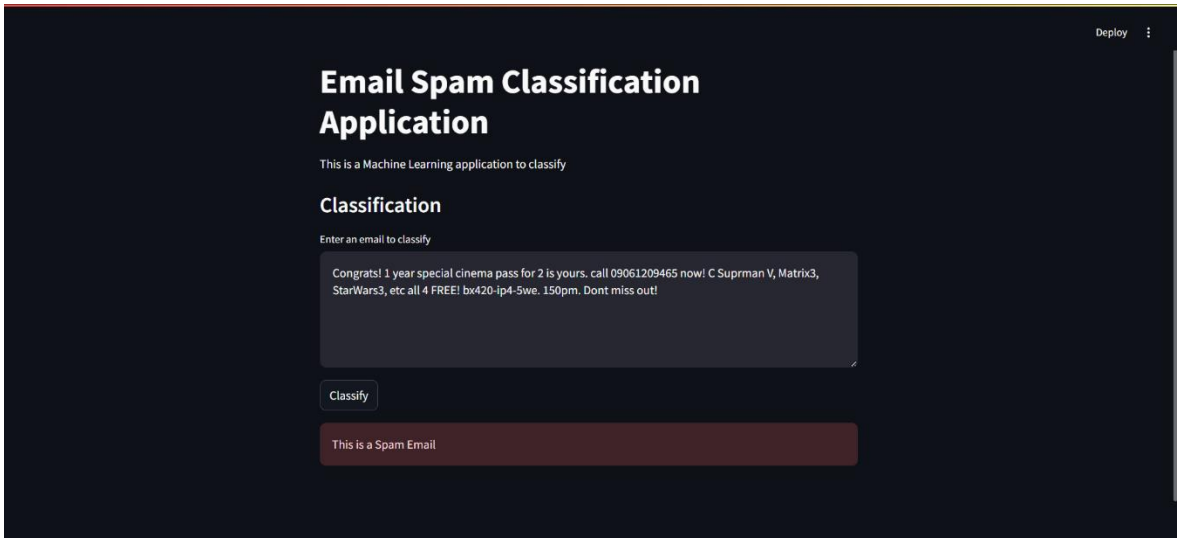

FIG 1: FOR SPAM EMAIL

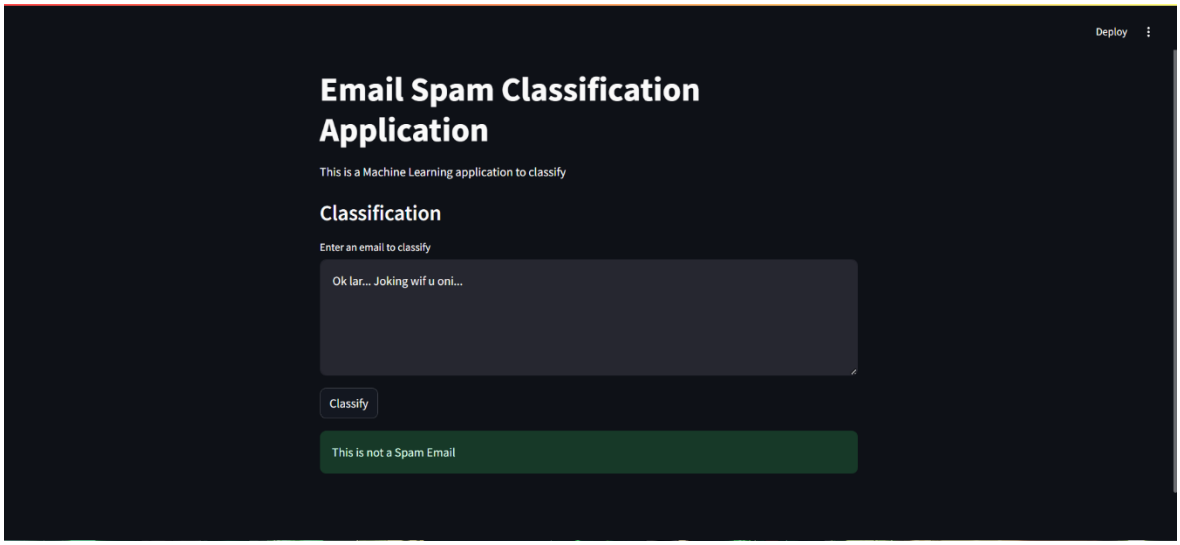

FIG 2: FOR NOT A SPAM EMAIL

The image appears to show a **Streamlit application** for **email spam classification**. Here's a breakdown of what it demonstrates:

Features of the Application:

1. **Title and Description**:
    - The app title is "Email Spam Classification Application".
    - A description explains that it's a machine learning application to classify emails.
2. **Input Section**:
    - There's a text box for users to enter an email they wish to classify.
    - The email input shown is a typical spam example containing keywords like "special offer", "call now", and "don't miss out".
3. **Action Button**:
    - A button labeled "Classify" is present. When clicked, it processes the input email.
4. **Output Section**:
    - The app identifies and classifies the input as "This is a Spam Email" with a red notification box.

4.2 GitHub Link for Code:

Spam-Email-Classification-using-NLP-and-Machine-Learning

CHAPTER 5

Discussion and Conclusion

5.1 Future Work:

While the developed spam email classification system achieves high accuracy, there are opportunities for further enhancement:

1. **Real-Time Classification**: Implement the model in a real-time pipeline for immediate email filtering.
2. **Deep Learning Integration**: Explore advanced models like Transformers (e.g., BERT) to improve contextual understanding of email text.
3. **Multi-Modal Analysis**: Incorporate non-textual features, such as email metadata (sender address, timestamps) and attachments, for comprehensive classification.
4. **Dynamic Learning**: Enable the system to adapt to evolving spam patterns through continuous learning with new data.
5. **Language Support**: Extend the system to support spam detection in multiple languages to cater to a global audience.
6. **Robustness against Adversarial Spam**: Enhance the model's resilience to sophisticated spam techniques, such as adversarial attacks and obfuscation.

5.2 Conclusion:

This project successfully demonstrates an efficient spam email classification system using Natural Language Processing (NLP) and Machine Learning (ML) techniques. By preprocessing email data, extracting relevant features, and leveraging supervised ML algorithms, the system achieves high accuracy in distinguishing spam from non-spam emails.

The project addresses the growing challenge of spam emails by providing a scalable, automated solution that can be implemented by email service providers or individual users. The robust performance of the chosen SVM model, combined with TF-IDF feature extraction, highlights the effectiveness of integrating traditional ML techniques with modern NLP approaches.

Overall, this work contributes to the domain of email filtering and cybersecurity, offering a foundation for future innovations. The suggestions for future work ensure the system remains adaptable to evolving email patterns and provides opportunities for continued improvement.

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]. Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.

[3]. Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python. O'Reilly Media.

[4]. Vaswani, A., et al. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30.