Mini Project Report on

Email/SMS Spam Classifier Report

Submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

Submitted by:

Student Name	University Roll No.
Anuj Rawat	*2218433*

Graphic Era Hill University

Department of Computer Science and Engineering
Graphic Era Hill University
Dehradun, Uttarakhand

January-2025

Graphic Era Hill University

DEHRADUN • BHIMTAL • HALDWANI

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the project report entitled **"Email/SMS Spam Classifier"** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering of the Graphic Era Hill University, Dehradun .

Name:-
Anuj Rawat

University Roll NO:-
2218433

Email/SMS Spam Classifier Report

Table of Contents

1. Introduction

Spam messages, often referred to as junk mail or unsolicited communications, are a prevalent issue in today's digital age. With the increasing use of email and SMS services, users are frequently bombarded with irrelevant or harmful content. The Email/SMS Spam Classifier project aims to address this challenge by providing an automated solution for spam detection. By leveraging Natural Language Processing (NLP) and Machine Learning (ML) techniques, the system classifies incoming messages into spam and non-spam categories, enabling users to focus on relevant and important communications.

This report delves into the details of the classifier, covering its design, implementation, evaluation, and potential future improvements.

2. Project Objectives

The primary objectives of this project include:

1. **Spam Detection:** Develop a reliable system to identify and filter spam messages.
2. **High Accuracy:** Ensure the classifier achieves superior performance metrics.

3. **Scalability:** Design the system to handle large volumes of messages.
4. **User-Friendliness:** Create an intuitive interface for seamless user interaction.
5. **Extendability:** Provide a framework for integrating advanced features in the future.

---

3. System Overview

The Email/SMS Spam Classifier processes user-submitted text messages, analyzes their content, and determines whether they are spam or non-spam. The system is built on three core components:

1. **Preprocessing Module:** Cleans and prepares text data for analysis.
2. **Feature Extraction:** Converts text into numerical representations using the TF-IDF technique.
3. **Prediction Model:** Utilizes a trained machine learning model to classify messages.

4. Features

Key Features

1. **Intuitive Interface:**
    - A user-friendly web interface built with Streamlit.
    - Allows users to input text messages and view results instantaneously.
2. **Real-Time Predictions:**
    - Processes input text and delivers classification results in real time.
3. **High Accuracy:**
    - Employs a robust ML model for reliable spam detection.
4. **Pre-Trained Model:**
    - Utilizes a pre-trained TF-IDF vectorizer and machine learning classifier.
5. **Lightweight Application:**
    - Optimized for deployment on systems with limited computational resources.

5. Implementation Details

Preprocessing Steps

Effective text preprocessing is crucial for improving the performance of NLP models. The following steps are implemented in the transform_text function:

1. **Lowercasing:** Converts all characters in the text to lowercase.
2. **Tokenization:** Splits the text into smaller units (tokens) using wordpunct_tokenize.
3. **Non-Alphanumeric Filtering:** Removes tokens containing special characters or punctuation.
4. **Stopword Removal:** Eliminates commonly used words like "the," "and," and "is" that do not add value.
5. **Stemming:** Reduces words to their base forms (e.g., "running" to "run") using the Porter Stemmer.

Machine Learning Model

The system employs multiple machine learning models to determine the best classifier for spam detection. These include:

1. **Logistic Regression**
2. **Random Forest**
3. **Bernoulli Naive Bayes (NB)**
4. **Multinomial Naive Bayes (NB)**
5. **Gaussian Naive Bayes (NB)**
6. **Extra Trees Classifier**

	Algorithm	Accuracy	Precision
1	KN	0.896518	1.000000
2	NB	0.962282	1.000000
5	RF	0.975822	1.000000
8	ETC	0.981625	1.000000
10	xgb	0.980658	0.975806
0	SVC	0.971954	0.973913
9	GBDT	0.951644	0.968085
6	AdaBoost	0.970986	0.957627
4	LR	0.952611	0.940594
7	BgC	0.965184	0.886364
3	DT	0.938104	0.862745

Model Selection Process

Each model was trained and evaluated using the same dataset to ensure a fair comparison. The Extra Trees Classifier emerged as the best-performing model with the following metrics:

- **Accuracy:** 98.16%
- **Precision:** 1 (perfect precision)

This model was selected for deployment due to its superior performance in both precision and overall accuracy.

Workflow

1. User inputs a text message.
2. The transform_text function preprocesses the input.
3. TF-IDF vectorization transforms the preprocessed text.
4. The machine learning model predicts the message's category (spam or non-spam).
5. Results are displayed on the user interface.

6. Tools and Technologies

- **Programming Language:** Python
- **Frameworks and Libraries:**
    - **Streamlit:** For building the user interface.
    - **NLTK:** For natural language preprocessing.
    - **Scikit-learn:** For training and deploying machine learning models.
    - **Pickle:** For model serialization.

---

7. Data Preparation

The quality and quantity of data play a pivotal role in determining the classifier's effectiveness. For this project:

1. **Dataset Source:**
    - A labeled dataset containing SMS messages was used, categorizing them as spam or non-spam.
2. **Data Cleaning:**
    - Removed irrelevant entries and duplicate records.
3. **Data Splitting:**
    - Divided the dataset into training and testing subsets in an 80-20 ratio.
4. **Feature Engineering:**
    - Applied TF-IDF to represent text data numerically.

5. **Handling Imbalance:**
    - Addressed data imbalance using oversampling techniques like SMOTE.

---

8. Performance Evaluation

Evaluation Metrics

1. **Accuracy:** Measures the overall correctness of the classifier.
2. **Precision:** Evaluates the proportion of correctly identified spam messages.
3. **Recall:** Assesses the ability to identify actual spam messages.
4. **F1-Score:** Balances precision and recall.

Results

Among the various models evaluated, the Extra Trees Classifier provided the best results:

- **Accuracy:** 98.16%
- **Precision:** 100%
- **Recall:** 96%
- **F1-Score:** 97.95%

These results indicate the classifier's exceptional performance, particularly in minimizing false positives.

9. Challenges Faced

Data Challenges

- **Imbalanced Dataset:**
    - Majority of messages were non-spam, requiring techniques like oversampling.

Model Challenges

- **Hyperparameter Tuning:**
    - Required extensive experimentation to optimize parameters.

Preprocessing Challenges

- **Handling Variability:**
    - Variations in text formatting and language required a flexible preprocessing pipeline.

10. Ethical Considerations

1. **Data Privacy:**
    - Ensured all data used was anonymized and handled securely.
2. **Bias Mitigation:**
    - Evaluated model outputs to reduce biases in spam classification.
3. **Responsible Deployment:**
    - Avoided misuse of the classifier for unethical purposes, such as profiling users.

11. Future Enhancements

1. **Integration with Messaging Platforms:**
    - Implement the classifier in email and SMS services.
2. **Advanced Algorithms:**
    - Explore deep learning models like BERT for improved results.
3. **Language Support:**
    - Extend the system to support multiple languages.
4. **Custom Filters:**
    - Allow users to set specific rules for spam detection.
5. **Mobile Compatibility:**
    - Develop a mobile app version of the application.

12. Conclusion

The Email/SMS Spam Classifier effectively demonstrates the use of NLP and ML for spam detection. Its robust design and high accuracy make it a valuable tool for managing digital communication. With ongoing advancements and enhancements, the system can evolve to address more complex challenges in spam classification.

13. References

1. Streamlit Documentation: https://docs.streamlit.io/
2. NLTK Library: https: //www.nltk.org/
3. Scikit-learn Library: https://scikit-learn.org/
4. Pickle Module: https://docs.python.org/3/library/pickle.html
5. TF-IDF Explanation: https://en.wikipedia.org/wiki/Tf–idf
6. SMOTE Technique : https://imbalancedlearn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html