MINOR PROJECT REPORT

SMS SPAM DETECTION

Submitted by

JISHNA.M

Mail id:jishna.mr@gmail.com

Index

Abstract

SMS spam detection is a critical aspect of maintaining the integrity and usability of mobile messaging systems. This project focuses on developing an efficient and accurate system for identifying and filtering spam messages from legitimate ones. Leveraging advanced machine learning algorithms and natural language processing techniques, the system analyzes the textual content of SMS messages to classify them into spam or non-spam categories.

The project involves several key components: data collection, preprocessing, feature extraction, model training, and evaluation. Initially, a diverse dataset of SMS messages is gathered and preprocessed to handle noise and inconsistencies. Features relevant to spam detection, such as keyword frequency, message length, and structural patterns, are extracted. Various machine learning models Naive Bayes, and machine learning approaches, are trained and tested to determine their effectiveness in spam detection.

The performance of these models is evaluated using metrics such as accuracy, precision, recall, and F1-score. The most effective model is then deployed in a real-time SMS filtering system, capable of distinguishing between spam and legitimate messages with high accuracy. The project aims to enhance user experience by reducing spam-related disruptions and improving the overall efficiency of mobile communication systems.

Introduction

SMS spam detection is the process of identifying and filtering unsolicited and often harmful text messages, commonly referred to as "spam," from legitimate or "ham" messages. This task is crucial for maintaining the quality of communication and protecting users from potential fraud, phishing attempts, and unwanted advertisements.

Key Concepts

1. **Spam**: Unwanted, unsolicited messages typically sent in bulk for commercial purposes. They can include promotional offers, scams, phishing attempts, or malicious content.
2. **Ham**: Legitimate messages that are desired and expected by the recipient. They include personal communication, important notifications, and other relevant messages.

Spam messages can clutter inboxes and detract from the user experience by overwhelming legitimate communications. Spam messages often contain links to phishing sites or malware. Effective spam detection helps protect users from potential security threats. Filtering out spam reduces the load on messaging systems and saves data usage and storage.

SMS spam detection is a critical component of modern communication systems, balancing user convenience, security, and effective resource management. As spam tactics evolve,

ongoing research and advancements in machine learning and NLP continue to improve spam detection capabilities.

Objective

To develop a classification model that accurately identifies SMS messages as either "ham" or "spam" using the SMS Spam Collection Dataset. The model will help automate the filtering of spam messages in real-world applications.

Future Directions

1. **Enhanced Models**: Using advanced machine learning techniques and pre-trained language models to improve accuracy and adaptability.
2. **Adaptive Systems**: Developing models that can dynamically adjust to new types of spam.
3. **Multilingual Detection**: Creating systems capable of detecting spam in various languages and dialects.
4. **Personalization**: Tailoring spam filters based on individual user preferences and behavior.

Tools and Environment

- **Programming Language: Python**
- **Development Environment: Jupyter Notebook**
- **Libraries:**
  - **pandas for data manipulation**
  - **numpy for numerical operations**
  - **scikit-learn for machine learning**
  - **nltk for natural language processing**
  - **matplotlib and seaborn for data visualization**

Techniques for Spam Detection

1. Rule-Based Systems:

A rule-based system in SMS spam detection utilizes predefined rules to classify messages as either "spam" or "ham" (legitimate). Unlike machine learning approaches that learn patterns from data, rule-based systems rely on explicitly defined criteria to make classifications.

- **Regular Expressions: Using patterns to match and filter out known spam structures.**

2. Machine Learning Models:

- **Supervised Learning: Training models on labeled datasets where messages are tagged as spam or ham. Common algorithms include:**

  .Logistic regression

3. Feature Extraction:

Feature extraction is a critical step in transforming raw text data into a numerical format that machine learning algorithms can process. In the context of SMS spam detection, feature extraction involves converting text messages into a set of features that represent the content of the messages in a way that can be used for classification.

- **Bag-of-Words (BoW): Represents text as a set of word counts or frequencies.**
- **Term Frequency-Inverse Document Frequency (TF-IDF): Measures the importance of words in a document relative to the entire corpus.**

4. Natural Language Processing (NLP) Techniques:

- **Text Preprocessing: Includes tokenization, stemming, lemmatization, and stopword removal.**
- **Semantic Analysis: Understanding the context and meaning of words in a message.**

Logistic regression

Logistic regression is a fundamental statistical method used for binary classification tasks. It predicts the probability that a given input belongs to a certain class .

Term Frequency-Inverse Document Frequency

Measures the importance of words in a document relative to the entire corpus. TF-IDF helps to account for the fact that some words appear frequently across all documents (e.g., "the") and might not be as informative.

Process:

- Term Frequency (TF): Measures how frequently a word appears in a document.
- Inverse Document Frequency (IDF): Measures how important a word is by considering its frequency across all documents.
- TF-IDF Score: Combines TF and IDF to give a weighted importance score for each word in each document.

Dataset Overview

- **Total Messages**: 5,574
- **Labels**:
  - **Ham**: 87% (4,844 messages)
  - **Spam**: 13% (730 messages)
- **Columns**:
  - **label**: Label (either "ham" or "spam")
  - **messages**: Raw text of the SMS message

Data Sources:

The dataset is compiled from multiple sources:

1. **Grumble Text Website**: 425 spam messages manually extracted.
2. **NUS SMS Corpus (NSC)**: 3,375 randomly chosen ham messages collected by volunteers from the National University of Singapore.
3. **Caroline Tag's PhD Thesis**: 450 ham messages.
4. **SMS Spam Corpus v.0.1 Big**: 1,002 ham messages and 322 spam messages.

Data Breakdown:

- **Ham Messages**: These represent legitimate, non-spam messages. Examples include personal communications or transactional notifications.

- **Spam Messages**: These include unsolicited messages such as advertisements or fraudulent offers.

Example Messages:

- **Ham**: "Ok lar... Joking wif u oni..."
- **Spam**: "Free entry in 2 a weekly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)"

5. Usage and Applications

1. **Research**: The dataset is used to develop and test machine learning models for SMS spam detection. Researchers can utilize it to benchmark the performance of different algorithms.
2. **Learning and Application**: It serves as a practice dataset for NLP techniques such as tokenization, stemming, and lemmatization. It also helps in understanding text classification problems.
3. **Model Benchmarking**: It provides a standard benchmark for evaluating the performance of various classification models in distinguishing between spam and legitimate messages.

6. Methodology

To leverage this dataset effectively:

1. **Data Preprocessing**: Includes text normalization, tokenization, and possibly feature extraction (e.g., TF-IDF).

2. **Model Training**: Use supervised learning techniques such as logistic regression, decision trees, or advanced models like support vector machines (SVM) and deep learning approaches.
3. **Evaluation**: Metrics such as precision, recall, F1-score, and accuracy are employed to assess model performance.

Scikit-learn:

Scikit-learn is a popular machine learning library in Python. It provides various tools for data preprocessing, modeling, and evaluation.**Scikit-learn** is a widely-used Python library for machine learning that provides simple and efficient tools for data analysis and modeling. It includes functions and classes for various tasks such as classification, regression, clustering, dimensionality reduction, and model selection.

train_test_split Function:

The train_test_split function is a utility provided by Scikit-learn to split a dataset into training and testing subsets. This function is crucial for evaluating the performance of machine learning models in a fair and unbiased manner.

Training (Logistic regression):

Logistic regression is a fundamental statistical method used for binary classification tasks. It predicts the probability that a given input belongs to a certain class

Results:

- **Accuracy**: 0.97
- **Precision, Recall, F1-score**: Detailed in the classification report.

Model Evaluation:

matplotlib.pyplot is a module from the Matplotlib library, which is a popular Python library for creating static, animated, and interactive visualizations.

MATLAB-like interface for creating plots and visualizations in Python. It is widely used for data visualization tasks, such as plotting graphs, histograms, scatter plots, and more.

Common Functions:

- **plt.plot()**: Plots lines and/or markers on a 2D plot.
- **plt.scatter()**: Creates scatter plots.
- **plt.hist()**: Generates histograms.
- **plt.bar()**: Creates bar charts.
- **plt.xlabel(), plt.ylabel(), plt.title()**: Set labels and titles for the plot.
- **plt.show()**: Displays the plot.

sklearn.metrics.ConfusionMatrixDisplay:

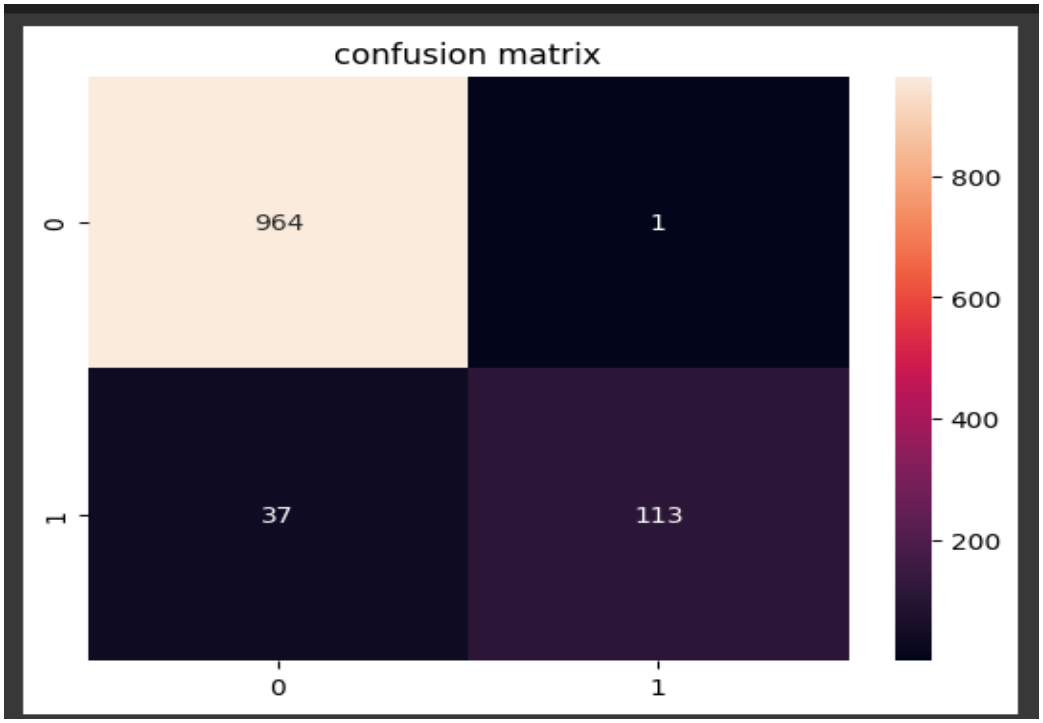
ConfusionMatrixDisplay is a class in the sklearn.metrics module of Scikit-learn. It is used to display confusion

matrices, which are useful for evaluating the performance of classification models.

Overview

- **Purpose**: Visualizes a confusion matrix, which helps in understanding how well a classification model performs by showing the counts of true positive, true negative, false positive, and false negative predictions.
- **Confusion Matrix**: A table layout that visualizes the performance of a classification model. It shows the actual vs. predicted classifications and helps identify misclassifications.

Visualize Result:



True Positives (TP): The number of correctly predicted spam messages.

True Negatives (TN): The number of correctly predicted legitimate messages.

False Positives (FP): The number of legitimate messages incorrectly classified as spam.

False Negatives (FN): The number of spam messages incorrectly classified as legitimate

Overall performace:

	precision	recall	f1-score	support
ham	0.96	1.00	0.98	965
spam	0.99	0.75	0.86	150
accuracy			0.97	1115
macro avg	0.98	0.88	0.92	1115
weighted avg	0.97	0.97	0.96	1115

summary

Based on the confusion matrix, the model's performance can be assessed through several key metrics:

1. **Accuracy: The overall accuracy of the model is high, suggesting that the model is generally effective in distinguishing between spam and legitimate messages.**
2. **Precision: The precision of the model indicates how reliably it classifies messages as spam. A high precision suggests that when the model predicts a message as spam, it is often correct.**
3. **Recall: The recall metric reflects the model's ability to identify all actual spam messages. A high recall value indicates that the model captures most of the spam messages, minimizing false negatives.**
4. **F1 Score: The F1 score balances precision and recall, providing a single metric that accounts for both false positives and false negatives. A high F1 score suggests a well-balanced model that performs well in both identifying spam messages and avoiding misclassification of legitimate ones.**

   The results provided are from a classification report, likely for a spam detection model. Here's a summary of the performance metrics:

1. **Class Performance:**
   - **Ham (Non-Spam):**

- **Precision: 0.96 – The model correctly identified 96% of the messages classified as ham.**
- **Recall: 1.00 – The model identified 100% of the actual ham messages.**
- **F1-Score: 0.98 – The harmonic mean of precision and recall, indicating excellent performance for ham classification.**
- **Support: 965 – The number of actual ham messages in the dataset.**

- **Spam:**
  - **Precision: 0.99 – The model correctly identified 99% of the messages classified as spam.**
  - **Recall: 0.75 – The model identified 75% of the actual spam messages.**
  - **F1-Score: 0.86 – The harmonic mean of precision and recall for spam, which is good but not as high as for ham.**
  - **Support: 150 – The number of actual spam messages in the dataset.**

2. Overall Performance:

- **Accuracy: 0.97 – The model correctly classified 97% of all messages.**
- **Macro Average:**

- **Precision: 0.98 – Average precision across both classes, treating each class equally.**
- **Recall: 0.88 – Average recall across both classes, treating each class equally.**
- **F1-Score: 0.92 – Average F1-Score across both classes, treating each class equally.**

- **Weighted Average:**
  - **Precision: 0.97 – Precision weighted by the number of instances in each class.**
  - **Recall: 0.97 – Recall weighted by the number of instances in each class.**
  - **F1-Score: 0.96 – F1-Score weighted by the number of instances in each class.**

Summary: The model performs very well overall, with high accuracy and strong performance in classifying ham messages. It has slightly lower recall for spam, which might suggest that it misses some spam messages, though precision for spam is excellent.

Challenges

1. **Evolving Nature of Spam**: Spammers continuously adapt their tactics, making it challenging to maintain effective filters.
2. **Class Imbalance**: Datasets often have many more ham messages than spam messages, leading to potential bias in models.
3. **Context Sensitivity**: Some messages may be ambiguous or context-dependent, complicating classification.
4. **Language and Culture**: Spam detection models need to handle variations in language and cultural contexts effectively.

Applications

1. **Mobile Phones**: Filtering spam messages directly on users' devices.
2. **Email Systems**: Integrated into email filters to block spam.
3. **Enterprise Communication**: Protecting organizational communication channels from spam and security threats.
4. **Customer Support**: Enhancing customer experience by reducing unwanted messages.