||Jai Sri Gurudev||

**BGSKH Education Trust**

# BGS COLLEGE OF ENGINEERING AND TECHNOLOGY

**MAHALAKSHMIPURAM, BENGALURU - 560086**

Branch : Artificial Intelligence & Machine Learning
Subject : Natural Language Processing
Subject Code : BAI601
Academic Year : 2024 - 25

# "SMS SPAM DETECTION"


**By:**

Name : Spandana
USN: 1MP22AI051
Name : Thanvi B
USN: 1MP22AI058

**Faculty:**

Dr Madhura Gangaiah
Professor, Dept. of AIML

# Abstract

With the widespread adoption of mobile technology, SMS (Short Message Service) has become one of the most common and convenient modes of communication. However, this popularity has also led to its exploitation by malicious actors who distribute spam messages to users at large scales. These spam messages often contain misleading information, promotional content, or phishing links that compromise user privacy and data security.

This project focuses on developing a robust SMS spam detection system using Natural Language Processing (NLP) and Machine Learning (ML) techniques. The goal is to classify SMS messages into two categories: **"spam"** and **"ham"** (legitimate). The proposed system uses a labeled dataset to train models such as **Naive Bayes**, **Logistic Regression**, and **Support Vector Machines (SVM)**. Key NLP preprocessing techniques—including tokenization, stop-word removal, stemming, and vectorization (TF-IDF/BOW)—are applied to prepare the data. The performance of each model is evaluated using metrics like **accuracy, precision, recall, and F1-score**.

The system is intended to be scalable, efficient, and capable of real-time deployment. Its implementation not only contributes to reducing unwanted message traffic but also enhances mobile security and user trust.

In recent years, the rapid growth of mobile communication has brought about significant benefits in accessibility and connectivity. SMS (Short Message Service) remains a dominant and cost-effective method for communication, especially in developing countries. Unfortunately, this convenience has also opened the door to abuse in the form of **spam messages**—unwanted or unsolicited messages sent in bulk, typically for advertising, phishing, or malicious purposes.

Spam messages are more than just an annoyance; they can lead to **data breaches**, **financial scams**, and **loss of trust** in mobile communication. Traditional methods of spam filtering, such as keyword blacklists or manual flagging, have proven inadequate in dealing with the evolving nature of spam content. This calls for intelligent, automated systems capable of learning from data and adapting to new spam tactics.

## Project Code

```
pip install pandas scikit-learn numpy

# Step 1: Import Libraries

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.feature_extraction.text import CountVectorizer

from sklearn.naive_bayes import MultinomialNB

from sklearn.metrics import accuracy_score, confusion_matrix, classification_report


# Step 2: Load Dataset

df =pd.read_csv('C:\\Users\\BGSCET\\Downloads\\sms+spam+collection\\SMSSpamCollection', sep='\t', header=None, names=['label', 'message'])

df.columns = ['label', 'message']

print(df.head())


# Step 3: Preprocess Data

df['label'] = df['label'].map({'ham': 0, 'spam': 1})  # Convert labels to binary


# Step 4: Split Data

X_train, X_test, y_train, y_test = train_test_split(df['message'], df['label'], test_size=0.2, random_state=42)
```

```
# Step 5: Convert Text to Vectors
vectorizer = CountVectorizer()
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)

# Step 6: Train Naive Bayes Classifier
model = MultinomialNB()
model.fit(X_train_vec, y_train)

# Step 7: Make Predictions and Evaluate
y_pred = model.predict(X_test_vec)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))

# Step 8: Predict Custom Message
sample = ["Congratulations! You've won a $1000 gift card. Call now!"]
sample_vec = vectorizer.transform(sample)
print("Prediction (1 = spam, 0 = ham):", model.predict(sample_vec)[0])
```

## Output

```
print(df.head())
```

```
  label                                            message
0   ham  Go until jurong point, crazy.. Available only ...
1   ham                      Ok lar... Joking wif u oni...
2  spam  Free entry in 2 a wkly comp to win FA Cup fina...
3   ham  U dun say so early hor... U c already then say...
4   ham  Nah I don't think he goes to usf, he lives aro...
```

```
Accuracy: 0.9919282511210762
Confusion Matrix:
 [[966   0]
 [  9 140]]
Classification Report:
              precision    recall  f1-score   support

           0       0.99      1.00      1.00       966
           1       1.00      0.94      0.97       149

    accuracy                           0.99      1115
   macro avg       1.00      0.97      0.98      1115
weighted avg       0.99      0.99      0.99      1115
```

```
sample = ["Congratulations! You've won a $1000 gift card. Call now!"]
sample_vec = vectorizer.transform(sample)
print("Prediction (1 = spam, 0 = ham):", model.predict(sample_vec)[0])
```

```
Prediction (1 = spam, 0 = ham): 1
```