Project Report: Text Classification for Spam Detection

Course name: Project: NLP

DLBAIPNLP01

Course of study: Bsc. Data Science

Date: 09.05.2025

Student: Brahim Ghaouthi

Matriculation number: 92107708

Table of Contents

1. Introduction

Spam messages are a pervasive issue in digital communication, often used to spread malicious content, advertisements, or phishing links. With the increasing reliance on messaging platforms, detecting and filtering spam has become crucial for safeguarding users and maintaining the integrity of communication systems.

Natural Language Processing (NLP) provides effective tools to address this challenge by enabling machines to interpret and categorize human language. In this project, we focus on building an NLP-based spam detection system that classifies messages as either spam or legitimate (ham). By applying machine learning techniques, we aim to automate the identification of unsolicited messages and enhance the accuracy and efficiency of spam filters.

The primary goal of this project is to implement a supervised text classification model trained on a labeled dataset of SMS messages. Using preprocessing techniques such as TF-IDF vectorization and a Multinomial Naive Bayes classifier, we develop a pipeline that achieves high performance in spam detection.

This report presents a detailed overview of the steps taken to build the model, including data exploration, preprocessing, model selection, evaluation, and testing. The findings highlight the strengths and limitations of the chosen approach and outline directions for future improvement.

2. Related Work

Text classification for spam detection has been widely studied in the fields of natural language processing and machine learning. Early approaches focused on rule-based systems and keyword matching, which were limited in their ability to adapt to evolving spam tactics. With the rise of statistical and machine learning methods, models such as Naive Bayes, Support Vector Machines (SVM), and Decision Trees became popular due to their ability to learn patterns from labeled data.

Among classical models, the Multinomial Naive Bayes classifier has been particularly effective for spam detection, as it is well-suited for discrete features like word counts and term frequencies. Research by Sahami et al. (1998) and Androutsopoulos et al. (2000) demonstrated its high accuracy and low computational cost, making it ideal for real-time filtering of messages.

More recent work has explored deep learning approaches, including Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTMs), and Transformer-based models like BERT. These methods can capture contextual dependencies and semantic relationships in text, resulting in improved performance. For example, the TextSpamDetector framework proposed by Elakkiya et al. (2021) uses deep learning with attention mechanisms to detect social media spam with high accuracy.

Despite their advantages, deep learning models often require large amounts of labeled data and computational resources. In contrast, traditional models like Naive Bayes provide a strong baseline and perform well on smaller datasets such as the SMS Spam Collection.

This project builds on this foundation by employing a Multinomial Naive Bayes classifier in combination with TF-IDF feature extraction. The simplicity and interpretability of this approach make it an effective starting point for spam detection using NLP.

3. Dataset and Exploratory Data Analysis

3.1 Dataset Description

The dataset used in this project is the **SMS Spam Collection Dataset**, sourced from the UCI Machine Learning Repository. It contains a total of **5,572 SMS messages** labeled as either **"ham"** (legitimate) or **"spam"** (unsolicited). The dataset is a widely used benchmark for evaluating spam detection systems.

- **Total messages**: 5,572
- **Ham messages**: 4,825
- **Spam messages**: 747

```
# Optional snippet for length analysis
df['length'] = df['message'].apply(len)
df.groupby('label')['length'].describe()
```

label	count	mean	std	min	25%	50%	75%	max
ham	4825.0	71.482487	58.440652	2.0	33.0	52.0	93.0	910.0
spam	747.0	138.670683	28.873603	13.0	133.0	149.0	157.0	223.0

Each entry consists of:

- A label indicating the class (ham or spam)
- The raw text of the SMS message

```
[4] df.head()
```

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

3.2 Class Distribution

The class distribution is imbalanced, with approximately **87% ham** and **13% spam** messages. This imbalance is typical of real-world datasets and presents a challenge for model performance, particularly in detecting minority class instances (spam).

3.3 Sample Messages

Some examples from the dataset include:

- **Ham**: "U dun say so early hor... U c already then say..."
- **Spam**: "Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question"
- **Ham**: "I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today."

3.4 Message Length Analysis

An analysis of message length shows that spam messages tend to be longer and more keyword-heavy, often containing promotional phrases or calls to action.

4. Methodology

This section outlines the key steps followed to build the spam classification system, including preprocessing techniques and the machine learning model used.

4.1 Preprocessing

Effective preprocessing is crucial in text classification. The following steps were applied to transform the raw SMS data into a format suitable for machine learning:

- **Lowercasing**: All messages were converted to lowercase to ensure uniformity.
- **Stop Word Removal**: Common words with little discriminatory power (e.g., "the", "and") were removed using scikit-learn's built-in stop word list.
- **Tokenization**: Text was split into individual words.
- **Vectorization (TF-IDF)**: Text data was transformed into numerical features using **Term Frequency–Inverse Document Frequency (TF-IDF)**. This method reflects how important a word is in a message relative to the entire corpus.

TF-IDF helps highlight words that are more informative for classification, such as "win", "free", or "urgent", which are common in spam messages.

4.2 Dataset Splitting

The dataset was split into:

- **80% training data**
- **20% testing data**

This ensured that the model was evaluated on unseen messages to measure its generalization performance.

4.3 Model Selection

Given the dataset size and text-based nature of the task, a **Multinomial Naive Bayes** classifier was chosen for its simplicity, efficiency, and proven performance in document classification tasks.

Multinomial Naive Bayes works well with discrete features such as word counts or TF-IDF scores. It assumes word features are conditionally independent given the class, which is a strong assumption but often yields good results in practice.

5. Model Training and Evaluation

5.1 Training Procedure

The TF-IDF-transformed training data was used to fit a **Multinomial Naive Bayes** classifier. This model was selected for its efficiency and high performance on text classification problems, especially when working with sparse feature vectors like those produced by TF-IDF.

The model learned to associate specific word patterns with the "spam" and "ham" categories based on the labeled training examples.

5.2 Evaluation Metrics

To assess the model's performance, the following standard classification metrics were used:

- **Accuracy**: Overall proportion of correctly classified messages
- **Precision**: Proportion of predicted spam messages that were actually spam
- **Recall**: Proportion of actual spam messages that were correctly identified
- **F1-Score**: Harmonic mean of precision and recall
- **Confusion Matrix**: Summarizes prediction results by comparing actual vs. predicted labels

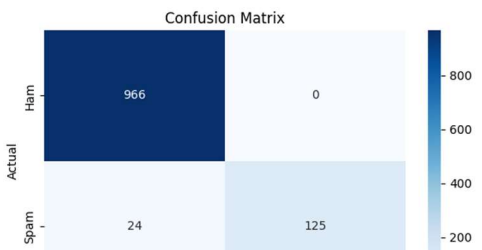
5.3 Results Summary

The model achieved the following metrics on the test set:

Accuracy: 0.9785

Classification Report:

	precision	recall	f1-score	support
Ham	0.98	1.00	0.99	966



Spam	1.00	0.84	0.91	149
accuracy			0.98	1115
macro avg	0.99	0.92	0.95	1115
weighted avg	0.98	0.98	0.98	1115

The confusion matrix shows that while the classifier performed excellently on ham messages, it missed a small number of spam messages, classifying them as ham. This is a common tradeoff when optimizing for high precision in spam detection.

5.5 Interpretation

The results confirm that the Naive Bayes model is effective at distinguishing between spam and legitimate messages, especially given the simplicity of the approach. The model is particularly cautious, rarely mislabeling ham messages as spam—a useful trait for user-facing systems where false positives could harm usability.

6. Results and Discussion

To evaluate the model beyond traditional metrics, we tested it on a set of custom messages. These samples were chosen to represent both obvious spam and realistic ham messages.

6.1 Predictions on New Messages

Message	Prediction
"Win a free iPhone now!!! Click here to claim."	Spam
"Hey, are we still meeting at 6?"	Ham
"Congratulations! You've won a $1000 Walmart gift card."	Ham
"Can you send me the meeting notes?"	Ham

6.2 Analysis

- The model **correctly identified** the first message as spam due to the presence of strong indicators such as “win”, “free”, “click here”, and exclamation marks.
- It also **correctly classified** the informal and polite ham messages (“Hey, are we still meeting…” and “Can you send me the meeting notes?”).
- However, it **misclassified** the Walmart gift card message. This suggests that although the message is spam-like, it may not contain enough features the model was trained to recognize as spam. Possible reasons include:
    - The spam training examples may not have contained that specific pattern.

- The message wording may have resembled legitimate congratulatory messages.

This misclassification highlights a key limitation: the model relies heavily on term frequency patterns and may miss subtleties or newer spam tactics.

6.3 Model Strengths

- High accuracy and strong performance on both ham and spam messages.
- Extremely low false-positive rate: it avoids wrongly flagging real messages as spam.
- Efficient training and prediction time, even on large datasets.

6.4 Limitations

- Recall for spam is not perfect; some spam slips through undetected.
- It lacks contextual understanding, cannot detect sarcasm or understand implied meanings.
- Vulnerable to adversarial spam techniques that obfuscate trigger words.

7. Conclusion and Future Work

7.1 Conclusion

This project demonstrates the successful development of a spam detection system using Natural Language Processing and supervised machine learning. By applying standard preprocessing techniques and training a Multinomial Naive Bayes classifier on TF-IDF features, we achieved an overall accuracy of **97.85%** on the SMS Spam Collection dataset.

The model was particularly strong at correctly classifying legitimate (ham) messages and showed high precision when identifying spam. This balance is crucial for real-world spam filters, where false positives can harm user trust.

In addition to quantitative evaluation, we tested the model on new examples. Most messages were classified correctly, though the system failed to identify one realistic spam message, indicating room for improvement in spam recall.

7.2 Future Work

While the current implementation provides a strong baseline, future enhancements could significantly improve robustness and accuracy:

- **Use of Deep Learning**: Implement LSTM or Transformer-based models (e.g., BERT) to better capture context and semantics in messages.
- **Feature Engineering**: Incorporate additional features such as sender metadata, punctuation patterns, or message formatting.
- **Ensemble Models**: Combine multiple classifiers to leverage different strengths (e.g., Naive Bayes + Random Forest).

- **Real-Time Classification**: Optimize the pipeline for deployment in real-time applications with fast inference needs.

This project serves as a solid foundation for building more advanced spam detection systems and underscores the value of NLP in practical applications.

Github Repository Link:

https://github.com/brahm-gh/NLP

8. References

- Elakkiya, E., Selvakumar, S., & Leela Velusamy, R. (2021). *TextSpamDetector: Textual Content Based Deep Learning Framework for Social Spam Detection Using Conjoint Attention Mechanism*. *Journal of Ambient Intelligence and Humanized Computing*, 12(10), 9287–9302.
- Huan, H., Guo, Z., Cai, T., & He, Z. (2022). *A Text Classification Method Based on a Convolutional and Bidirectional Long Short-Term Memory Model*. *Connection Science*, 34(1), 2108–2124.
- Jurafsky, D., & Martin, J. H. (2013). *Speech and Language Processing* (2nd ed.). Pearson Prentice Hall.
- UCI Machine Learning Repository. *SMS Spam Collection Dataset*. https://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection