Text Mining Project: SMS Spam Classification and Clustering

Pasquale Gravante Antonio Mastroianni

January 18, 2024

1 Introduction

The large presence of mobile devices and the widespread use of SMS for communication, have led to an escalating challenge of dealing with unwanted messages, particularly spam. Text mining plays an important role in addressing this issue. This report is a comprehensive exploration of the application of text mining techniques for the classification and clustering of SMS messages, with a specific focus on the detection of spam.

The prevalence of mobile communication has put a focus on the importance of developing robust methods for filtering out spam messages. The ability to accurately classify and cluster SMS messages not only enhances user experience by minimizing unwanted intrusions but also holds significant implications for security and privacy. In this project, we aim to tackle the challenges posed by SMS spam through the application of text mining methodologies.

The main objectives of this project are two: first, to build a reliable SMS spam classifier capable of accurately distinguishing between legitimate and spam messages; second, to explore clustering techniques that group similar messages together, providing additional insights into the underlying patterns and characteristics of SMS data. By achieving these goals, we aim to contribute to the broader field of text mining and enhance our understanding of SMS communication dynamics.

2 Problem Description

The project addresses the pressing issue of SMS spam within the broader context of text mining. SMS spam poses a unique challenge due to the limited length of messages and the need for efficient and accurate detection methods. In this section, we provide a detailed description of the problem, outlining the characteristics of spam messages and the potential impact on users and mobile networks.

Challenges in SMS spam detection arise from the brevity of messages, varied content, and attempts to deceive recipients. Traditional spam detection methods used in email filtering may not directly translate to SMS. Understanding these challenges is crucial for developing effective text mining solutions tailored to the specific nuances of SMS communication.

Accurate detection of SMS spam is essential to safeguard users from unwanted content and potential security threats. False positives and false negatives can have significant consequences, underscoring the need for robust classification and clustering techniques. The outcomes of this project aim to contribute to the advancement of methods that strike a balance between precision and recall in SMS spam detection.

3 Data Description and Exploratory Data Analysis

This section is dedicated to comprehending the characteristics of the dataset used in the project. Through exploratory data analysis (EDA), we aim to unveil inherent patterns and gain insights into the dataset's composition.

3.1 Dataset overview

The SMS Spam Collection comprises a set of SMS tagged messages, meticulously assembled for SMS spam research. The dataset consists of 5,574 SMS messages, where each row represents a distinct message. There are two columns: one containing the raw text of the message and the other indicating the label, distinguishing between 'ham' (legitimate) and 'spam.'

3.2 EDA

In this section we are going to take a look about the target distribution and the messages length.

- **Message Classes**:

Class	Messages count
Ham	4527
Spam	728

Table 1: Message labels distribution

The table above provides an overview of the distribution of message labels. It is evident that the majority of messages are labeled as 'ham' (4527), indicating legitimate messages, while the 'spam' category consists of 728 messages, underlying the fact that we are working with unbalanced classes.

- **Message Length Distributions:**

- **Description:** The histograms that are shown below illustrate the distribution of messages, firstly in terms of characters length and then in terms of word length, trying to underly the main differences between the messages that belong to the spam class and the legitimate ones.

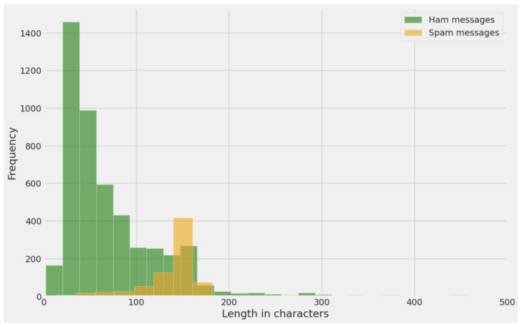


Figure 1: *Histogram of character length distribution in messages.*

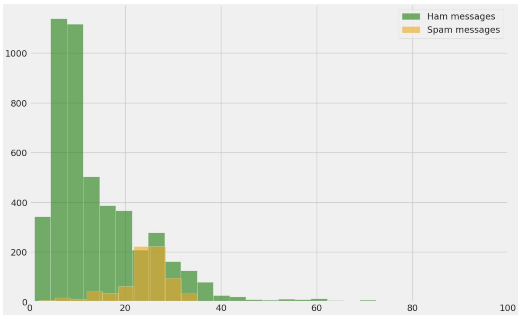


Figure 2: *Histogram of word length distribution in messages.*

The two histograms exhibit striking similarities, with a notable observation that spam messages generally tend to be longer than legitimate ones, which often appear to be concise and brief.

- **Word Clouds:** Word clouds were generated to see graphically what are the discriminatory words for each group of messages:

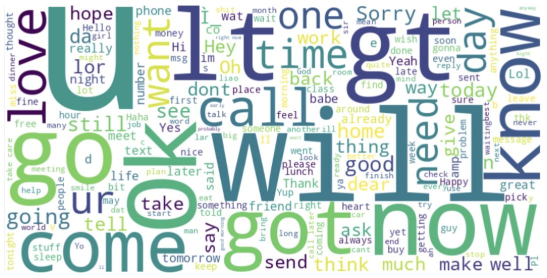


Figure 3: *Ham messages wordcloud.*

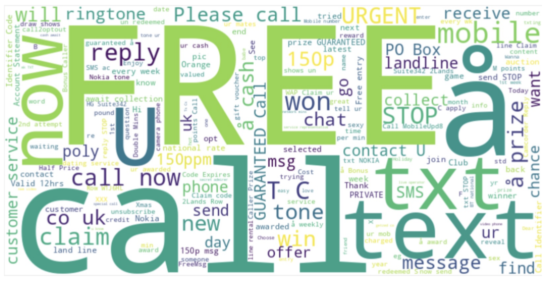


Figure 4: *Spam messages wordclouds.*

4 Data Pre-processing

This section outlines the sequential steps taken to prepare the SMS dataset for subsequent analysis, ensuring the data is refined and well-suited to apply the text-mining tasks.

4.1 Text Cleaning

Text cleaning is a crucial initial step to enhance the quality of the textual data. In order to achieve that, we performed the following operations on the raw text:

- **Converting to lowercase:** Standardize the text by converting all characters to lowercase, ensuring uniformity in the subsequent analyses.
- **Removing square brackets:** Eliminate text within square brackets, as it often contains non-essential or noisy information.
- **Removing links and HTML tags:** Clean the text by removing hyperlinks and HTML tags, ensuring only the text content is considered.
- **Removing punctuation:** To focus on content-carrying words, punctuation has been removed, resulting in a more accurate representation of the language.
- **Removing newlines and numbers:** Enhance cleanliness by eliminating newline characters and words containing numbers, streamlining the dataset.

4.2 Stopword Removal

Stopwords, common words that do not carry significant meaning on their own, have been removed in order to focus on content-rich words. This includes standard English stopwords as well as domain-specific stopwords, that is, words that are usually used as abbreviations in text messages, like: ’u’, ’im’, ’c’, ’ur’, ’dont’, ’doin’, and ’ure’.

4.3 Word Stemming

Word stemming, that has been executed through the *SnowballStemmer* in English, has been employed to reduce the dimensionality of the dataset. This process involves converting words to their base form, capturing the root meaning and improving the efficiency of the next analyses.

4.4 Label Encoding

Label encoding has been applied to the target variable ’label’ by using the *LabelEncoder*. This transformation assigns numerical values to the categorical labels (’ham’ and ’spam’), a prerequisite for most models that operate on numerical data.

4.5 Text Vectorization

In the text vectorization phase, our aim is to convert raw SMS messages into a format suitable for machine learning models. We employ two main techniques: Count Vectorization and TF-IDF (Term Frequency-Inverse Document Frequency) transformation.

- Count Vectorization is the initial step. It involves creating a matrix where each row corresponds to an SMS message, and each column represents a unique word in the entire dataset. Values in the matrix signify the frequency of each word in the respective message. Common stop words are excluded to focus on meaningful content, and we extend this process to capture word pairs (bigrams).

- Following Count Vectorization, we proceed to TF-IDF transformation. This enhances our numerical representation by considering not only local word frequency but also global significance. The TF-IDF matrix assigns weights based on a word's occurrence in a specific message and its rarity across the dataset. This identifies words distinctive to certain messages, contributing to an informative feature set.

- Simultaneously, we integrate pre-trained word embeddings using GloVe. These embeddings offer a dense representation of words in a continuous vector space, capturing semantic relationships. The pre-trained GloVe embeddings are loaded, creating a matrix where each row corresponds to a unique word, and columns represent embedding dimensions. This matrix initializes the embedding layer in subsequent neural network models, enriching word representation with semantic information.

- Importantly, we ensure uniformity in our sequences through padding. This involves adjusting the length of our sequences to a consistent size. Uniform length is crucial for creating consistent input shapes for machine learning models, as they typically require fixed-size input.

5 Classification

In this section different methods to classify our SMS messages are explored, with the aim of having as much accuracy as possible in discriminating between legitimate messages and spam ones.

5.1 Multinomial Naive Bayes Classifier

The first chosen classifier, the Multinomial Naive Bayes model, plays a pivotal role in categorizing messages into two distinct classes: 'ham' for legitimate messages and 'spam' for the following reasons:

- Naive Bayes models are particularly efficient when dealing with textual data, making them well-suited for tasks involving NLP. Text messages, being short and often containing distinctive keywords, align well with the assumptions of the Naive Bayes algorithm.

- Naive Bayes is a simple and fast algorithm, making it computationally efficient. This simplicity involves a quick training and prediction, especially in scenarios where real-time or near-real-time classification is desirable, like in this context.

- Naive Bayes performs well even with limited training data. In the context of SMS spam classification, where obtaining a vast labeled dataset might be challenging, like in our case, Naive Bayes can still provide reliable results.

5.1.1 Training Process

The classifier undergoes a crucial training phase, where it learns intricate patterns and associations from the provided feature matrix (train messages) and corresponding target labels. This step is essential for the model to make informed predictions.

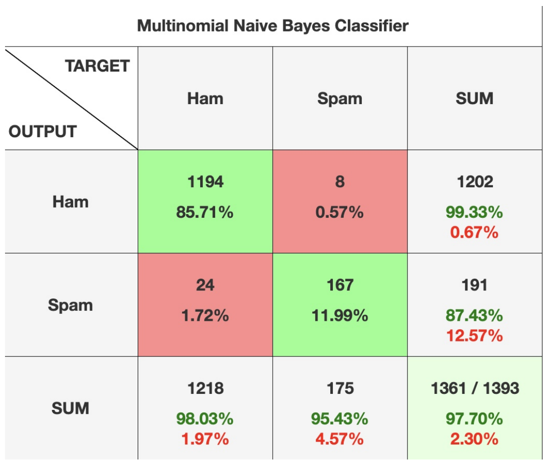
5.1.2 Predictive Analysis

Post-training, the Multinomial Naive Bayes classifier showcases its efficacy by accurately assigning class labels to new data, that is our test set. Beyond pure classifications, the model estimates the likelihood of each prediction, providing valuable insights into its confidence levels.

5.1.3 Assessment Metrics

The evaluation of the classifier's performance involves the application of standard metrics, with a particular emphasis on accuracy. Additionally, a detailed examination of the confusion matrix enhances our understanding of true positives, true negatives, false positives, and false negatives, shedding light on the model's strengths and areas for improvement.

To visually represent the classifier's performance, we include a graph illustrating the confusion matrix. This graphical representation offers a clear overview of the model's ability to correctly and incorrectly classify instances, providing a perspective on its overall performance.



Multinomial Naive Bayes Classifier

OUTPUT \ TARGET	Ham	Spam	SUM
Ham	1194 85.71%	8 0.57%	1202 99.33% 0.67%
Spam	24 1.72%	167 11.99%	191 87.43% 12.57%
SUM	1218 98.03% 1.97%	175 95.43% 4.57%	1361 / 1393 97.70% 2.30%

Figure 5: *Confusion matrix of Multinomial Naive Bayes Classifier.*

The high number of true positives and true negatives, together with low false positives and false negatives, underlies the classifier's robustness in distinguishing between 'ham' and 'spam' messages. These results confirm the reliability and effectiveness of the Multinomial Naive Bayes model in the context of SMS spam classification.

5.2 Logistic Regression

Logistic Regression emerges as a strategic choice for the classification of text messages within our SMS classification project. As a binary classification algorithm, Logistic Regression is well-suited to address our problem, where messages are categorized into spam or non-spam. Some of the reasons why it has been employed are:

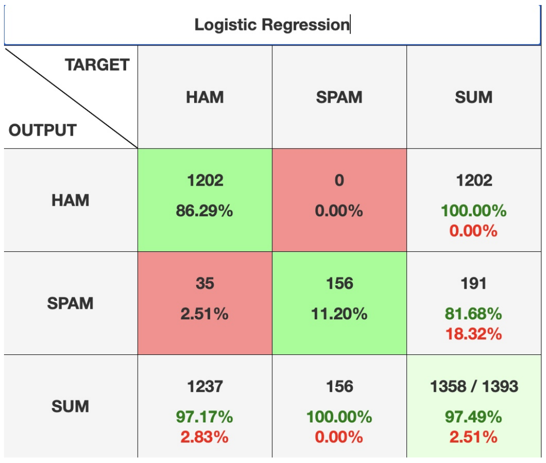
- **Interpretability:** In the context of distinguishing between spam and ham messages, interpretability is crucial. Logistic Regression offers transparent insights into the impact of individual features, allowing us to understand the factors influencing the likelihood of a message being classified as spam.
- **Linear Decision Boundary for Textual Patterns:** The linear decision boundary employed by Logistic Regression aligns with the underlying patterns in textual data. This simplicity ensures effective modeling of the relationships between words and the likelihood of a message being spam.
- **Probabilistic Output:** Logistic Regression outputs probabilities, providing a perspective on the likelihood of classifications. This probabilistic nature allows for flexible threshold setting, adapting to the specific requirements of spam detection in SMS.

5.2.1 Training and Testing

The Logistic Regression model is trained by maximizing the likelihood of the observed data. The goal is to find the optimal values for the weights (β) that maximize the likelihood of the actual class labels given the input features.

5.2.2 Testing

The Logistic Regression classifier was then evaluated on the test dataset, and its performance was assessed using various metrics. The confusion matrix provides a detailed breakdown of the classifier's predictions:



Logistic Regression

TARGET / OUTPUT	HAM	SPAM	SUM
HAM	1202 86.29%	0 0.00%	1202 100.00% 0.00%
SPAM	35 2.51%	156 11.20%	191 81.68% 18.32%
SUM	1237 97.17% 2.83%	156 100.00% 0.00%	1358 / 1393 97.49% 2.51%

Figure 6: *Logistic Regression Classification*

As we can see from the confusion matrix, the model is very good at identifying the legitimate messages but struggles a bit in detecting the spam ones, meaning that it has the maximum recall but a low precision, despite having a 97.50% accuracy.

5.3 BERT

This section delves into the utilization of a BERT-based model for SMS message classification, a state-of-the-art architecture renowned for its prowess in natural language processing tasks.

5.3.1 Model Configuration

The BERT model is configured with an Adam optimizer and a cross-entropy loss function. The training process spans three epochs, during which the model is fine-tuned to enhance its ability to discern between spam and legitimate messages.

5.3.2 Training Loop

The training loop iterates through the dataset, iteratively updating the model parameters to minimize the cross-entropy loss. Training losses and accuracy metrics are tracked, providing insights into the model's learning progress.

5.3.3 Validation Process

After each epoch, the model undergoes validation to assess its generalization performance on an independent dataset. Validation accuracy is a critical metric, gauging how well the model can classify unseen SMS messages.

5.3.4 Performance Visualization

To comprehend the training dynamics, visualizations are employed. The first plot illustrates the training loss over epochs, showcasing how the model converges. The second plot depicts the training accuracy, with a reference line indicating the validation accuracy after the initial epoch.

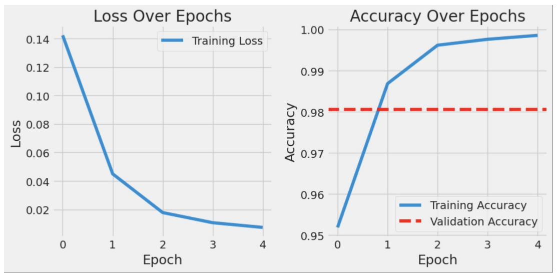
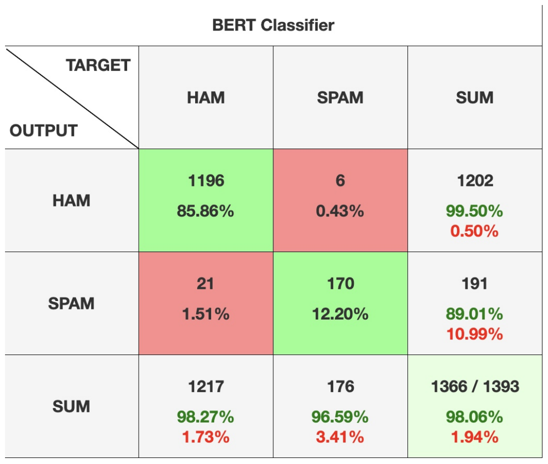


Figure 7: *Loss and Validation over epochs*

As shown by the figure, the loss function was decreasing with every epoch of training and the accuracy of the classification on the training test was also improving, whilst having a validation accuracy of approximately 0.98.

5.3.5 Outcome and Evaluation

Upon completing the training and validation processes, we show the confusion matrix of the classifications made by our model:



BERT Classifier

TARGET / OUTPUT	HAM	SPAM	SUM
HAM	1196 85.86%	6 0.43%	1202 99.50% 0.50%
SPAM	21 1.51%	170 12.20%	191 89.01% 10.99%
SUM	1217 98.27% 1.73%	176 96.59% 3.41%	1366 / 1393 98.06% 1.94%

Figure 8: *Confusion matrix of BERT Classifier*

The final validation accuracy stands at approximately 98.06%. This metric, serves as a key indicator of the model's efficacy in classifying SMS messages. The BERT-based classifier demonstrates notable success in distinguishing between spam and legitimate SMS messages, underscoring its potential for real-world deployment.

6 Clustering Analysis

In this section, we dive into the comprehensive analysis of the SMS dataset through the application of two distinct clustering algorithms: K-means and Hierarchical clustering. The goal is to uncover underlying patterns and group similar SMS messages together.

6.1 K-means Clustering

The initial step involved transforming the SMS messages into a TF-IDF (Term Frequency-Inverse Document Frequency) matrix, capturing the importance of each term in the corpus. The K-means clustering algorithm was then applied to identify natural groupings within the dataset. To determine the optimal number of clusters, a range from 2 to 40 clusters was explored.

6.1.1 Cluster Evaluation Metrics

Silhouette scores and Davies-Bouldin indices were employed as key metrics to evaluate the quality of clustering for different cluster counts. The silhouette score measures the compactness and separation of clusters, while the Davies-Bouldin index assesses the cluster dispersion. The following table summarizes the evaluation metrics:

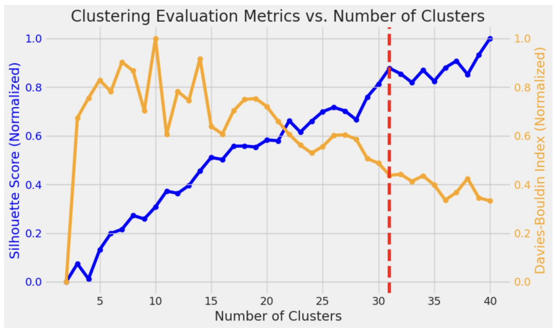


Figure 9: *Clustering evaluation metrics vs Number of Clusters.*

6.1.2 Optimal Cluster Number Selection

After a careful examination of the metrics, the optimal number of clusters was chosen as 31, where the trade-off between intra-cluster cohesion and inter-cluster separation was deemed most effective. This choice forms the basis for the subsequent analyses.

6.1.3 Cluster Analysis and Visualization

Each resulting cluster was then analyzed to identify distinctive patterns within the SMS messages. To provide a visual representation, word clouds were generated for the top terms in each cluster, shedding light on the prevalent themes and contexts within the identified groups.

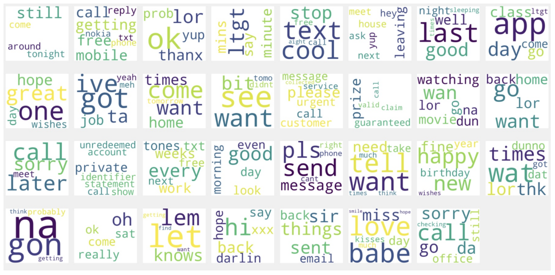


Figure 10: *Clusters wordclouds.*

6.2 Hierarchical Clustering

In addition to K-means, Hierarchical clustering was applied to further explore the hierarchical relationships within the SMS dataset. Ward's linkage method was employed for this purpose.

6.2.1 Dendrogram Cut Analysis

Hierarchical clustering results in a dendrogram, representing the hierarchy of clusters.
Various cuts were applied to the dendrogram to create different cluster configurations. Silhouette scores and Davies-Bouldin Index were computed for each cut to assess the quality of clustering.

The optimal cut was identified at a height of 13.8, resulting in 31 clusters, mirroring the choice made in the K-means analysis.

7 Results and Conclusions

In this section, the results obtained from the application of different classifiers to the task of SMS classification are presented. The classifiers include the BERT model, Multinomial Naive Bayes Classifier, and Logistic Regression.

7.1 Classification

To understand which classifier worked better, evaluation metrics such as accuracy and F1 score (i.e., the harmonic mean between precision and recall) are provided to give a comprehensive overview of their performance:

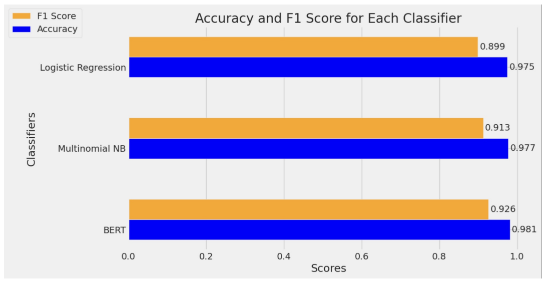


Figure 11: *Classifiers comparison*

As show by the figure, all classifiers provide a good classification results for this task, more specifically:

- **Logistic Regression**: It produces a classification with **97.5%** accuracy and an F1 score of **0.899**, given by a maximum **Recall** and a pretty low **Precision**, meaning that the model is not missing any spam messages (captures them all), but having low precision suggests that there is a higher likelihood of incorrectly labeling legitimate messages as spam.

- **Multinomial Naive Bayes Classifier**: It produces a classification with **97.77%** accuracy and an F1 score of **0.9126**, given by a more balanced result between precision and recall, the latter being a bit higher. This model is more solid with the respect to the previous one and it preferred in this context.

- **BERT Classifier**: It produces a classification with **98.06%** accuracy and an F1 score of **0.9264**, suggesting a very balanced result between precision and recall, and so a very efficient classifier.

In conclusion, the BERT model, also thanks to the pre-trained contextual embeddings, proved to be the most effective in handling the complexities of natural language in the SMS domain, but in a context without the addition of pre-trained word embeddings, the Multinomial Naive Bayes Classifier looks like a very suitable alternative.

7.2 Clustering

The clustering analysis revealed nuanced insights into the structure of the SMS dataset. The selection of K-means with 31 clusters and Hierarchical clustering with a specific cut height of 13.8 facilitated the identification of meaningful patterns and groupings within the messages. We can see it graphically:

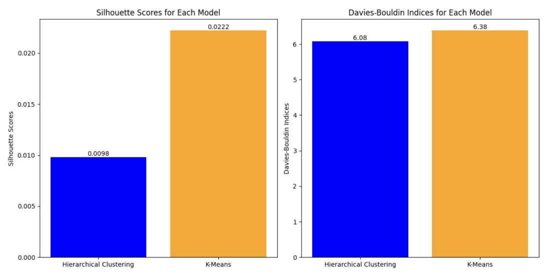


Figure 12: *Indices comparison*

As we can see from the plot, the Davies-Bouldin index, that has to be minimized, is similar for both models, while for what concerns that Silhouette scores, it is suggested by the higher value that K-means algorithm generated slightly better clusters of messages,implying that the clusters are better defined when using K-means algorithm.