TACKLING UNWANTED TEXTS
Filtering Spam SMS Messages

Pranav Reddy Bande
Nilay N Patel

Computer Science
Georgia State University

Abstract

Abstract - With the escalating influx of SMS-based marketing and fraudulent activities, the need for a reliable system to differentiate between genuine and unwanted messages has become imperative. Traditional manual filtering methods are inefficient and error-prone, necessitating the development of an automated solution. This project aims to devise and implement an intelligent system capable of accurately identifying spam messages and preventing them from reaching users' inboxes. Leveraging advanced machine learning algorithms and natural language processing techniques, the proposed system will analyze SMS content and metadata to distinguish between legitimate and spam messages with high precision. By addressing this critical challenge, the project seeks to enhance user experience, reduce the impact of unwanted messages, and fortify the integrity of communication channels.

1. Introduction

1.1 Background

In the contemporary landscape of digital communication, the pervasive influx of Short Message Service (SMS) messages has necessitated robust mechanisms for filtering and cleaning unwanted content. This project endeavors to address this imperative by creating a sophisticated machine-learning model that discerns between legitimate (HAM) and spam SMS messages using Natural Language Processing (NLP) techniques. Leveraging advanced features engineered from textual data, such as word count, presence of currency symbols, and numerical content, the model aims to accurately classify incoming SMS messages, thereby facilitating effective filtering and decluttering of users' SMS inboxes.

1.2 Objectives

The primary objective of this project is to develop a machine-learning model capable of automatically detecting and classifying SMS messages as either spam or normal (HAM). By leveraging NLP techniques, the model will analyze the textual content of incoming messages and extract relevant features indicative of spam behavior, such as unusual word patterns, the presence of promotional language, and syntactic anomalies. Through the engineering of key features like word count, presence of currency symbols, and numerical content, the model will enhance its ability to discern between genuine and unwanted messages, thereby facilitating efficient SMS filtering and decluttering.

1.3 Significance and Impact

The successful implementation of this project holds significant implications for enhancing user experience and communication integrity in the realm of SMS-based communication. By enabling automated filtering and cleaning of SMS messages directly on users' devices, the project aims to alleviate the burden of sifting through unwanted content manually. This, in turn, not only saves users' time and effort but also mitigates the potential risks associated with falling victim to spam or fraudulent SMS campaigns. Moreover, the availability of such a robust filtering mechanism contributes to fostering trust and confidence among users, thereby fortifying the integrity of SMS communication channels.

1.4 Resource Utilized

This project harnesses a diverse array of resources, including Python programming libraries such as

pandas, numpy, sklearn, matplotlib, seaborn, and nltk. Additionally, the dataset utilized for model training and evaluation is sourced from the UCI Machine Learning repository on Kaggle. The dataset comprises a comprehensive collection of labeled SMS messages, encompassing both spam and normal (HAM) categories, thereby providing a robust foundation for model development and evaluation. Through the judicious utilization of these resources, the project aims to deliver a scalable and efficient solution for SMS filtering and decluttering, thereby enhancing communication security and user experience.

Python Programming Libraries:

- pandas: For data manipulation and analysis.
- numpy: For numerical computations and array operations.
- sklearn: For implementing machine learning algorithms and model evaluation.
- matplotlib and seaborn: For data visualization and graphical representation of results.
- nltk (Natural Language Toolkit): For natural language processing tasks such as tokenization, stemming, and stop-word removal.

Dataset by UCI Machine Learing on Kaggle: https://www.kaggle.com/uciml/sms-spam-collection-dataset

2. System Architecture

2.1 Data Collection and Preprocessing:
The system begins by loading the SMS dataset and performing preprocessing steps such as handling NaN values, converting text to lowercase, removing special characters and numbers, tokenizing, removing stopwords, lemmatizing, and building a corpus of messages.

2.2 Exploratory Data Analysis (EDA):
EDA is conducted to explore the dataset, visualize the distribution of SMS labels (Spam vs. Ham), and gain insights into feature distributions and imbalances.

2.3 Feature Engineering:
Imbalanced dataset handling is implemented using oversampling techniques to address class imbalance. New features such as word_count, contains_currency_symbol, and contains_number are created to enrich the dataset and improve model performance.

2.4 Data Cleaning: Further data cleaning steps involve removing special characters and numbers using regular expressions, converting text to lowercase, tokenizing, removing stopwords, lemmatizing words, and building a corpus of messages for modeling.

2.5 Model Building & Evaluation: Multiple classification models including Multinomial Naive Bayes, Decision Tree, Random Forest, and a Voting Ensemble are trained and evaluated using cross-validation. The F1-Score metric is utilized for model evaluation, with Random Forest achieving the highest performance.

2.6 Prediction:
Finally, the system allows for making predictions on new SMS messages using the trained Random Forest model to classify messages as spam or ham.

2.7 Exploring Model Variability and Performance Metrics
In the code implementation, we embark on a journey to developing a robust SMS filtering and decluttering system using machine learning and natural language processing techniques. The objective is clear: to automate the detection of spam messages and enhance user experience by filtering unwanted content from SMS inboxes.

Our approach is systematic and comprehensive. We'll start by preprocessing and engineering the dataset, ensuring it's primed for analysis. Then, we'll delve into the realm of exploratory data analysis, gaining insights into the dataset's composition and distribution of SMS labels.

Next comes the exciting part – model building. We'll explore a variety of classification algorithms, including

Multinomial Naive Bayes, Decision Tree, Random Forest, and a Voting Ensemble. By employing cross-validation, we'll rigorously evaluate each model's performance, scrutinizing metrics like accuracy, precision, recall, and F1-Score.

Our aim is simple yet ambitious: to identify the algorithm that best suits our task of spam detection. Through this process, we'll uncover the strengths and weaknesses of each model, paving the way for informed decisions in model selection.

3. Implementation

3.1 Data Loading and Preprocessing:

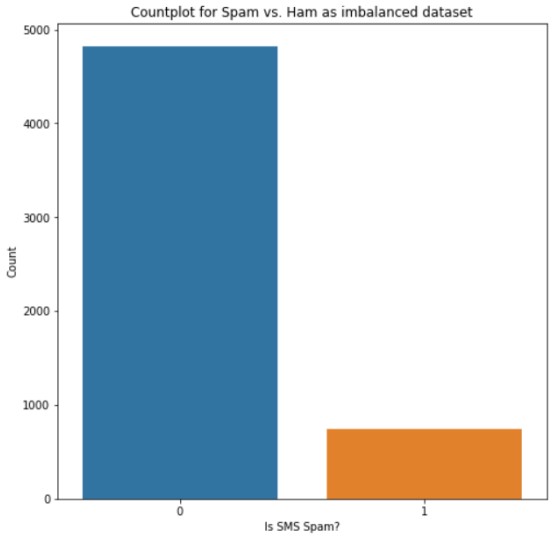
We begin by importing essential libraries such as pandas and numpy, followed by loading the SMS dataset into a pandas data frame. The dataset is then preprocessed to handle any NaN values and ensure consistency in data format.

3.2 Exploratory Data Analysis (EDA):

Using seaborn and matplotlib libraries, we visualize the distribution of SMS labels (Spam vs. Ham) through count plots. This step provides valuable insights into the dataset's composition and imbalance, setting the stage for further analysis.

3.3 Feature Engineering:

3.3.1 Imbalanced dataset handling: We implement oversampling techniques to address class imbalance, ensuring a more balanced representation of spam and ham messages in the dataset.



Insight: *From the above countplot, it is evident that the* ***dataset is imbalanced****.*

Figure:3.1

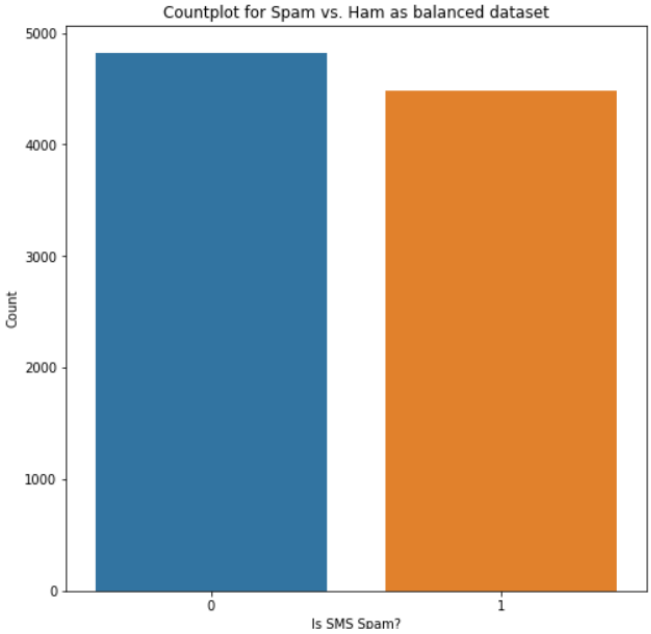
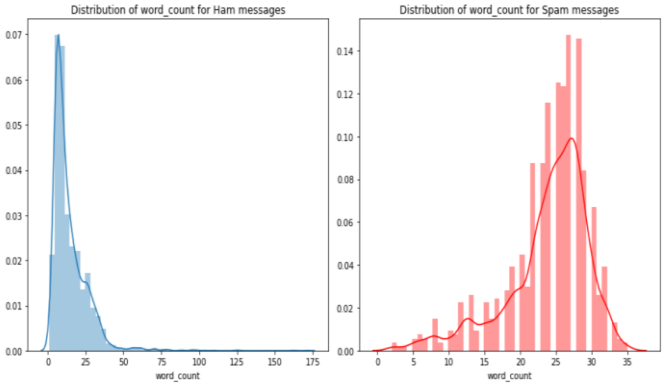


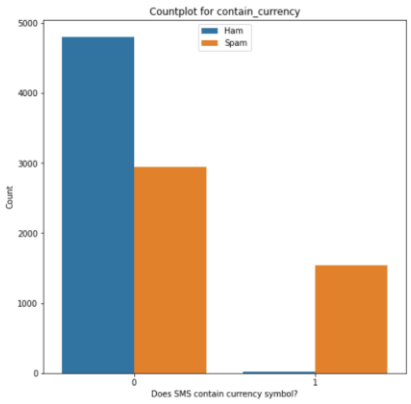
Figure:3.2

3.3.2 Creation of new features: Additional features such as word_count, contains_currency_symbol, and contains_number are engineered from existing features, enriching the dataset with valuable information for model training.



*Insight: **Spam messages** word_count fall in the range of **15-30 words**, whereas majority of the **Ham messages** fall in the range of **below 25 words**.*

Figure:3.3



*Insight: **Almost 1/3 of Spam messages contain currency symbols**, and currency symbols are **rarely used in Ham messages.***

Figure:3.4

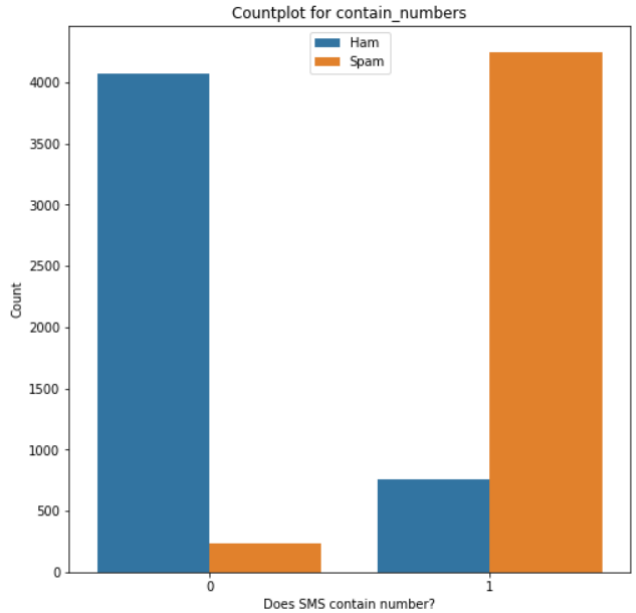


Figure:3.5

Figure:3.5 *It is evident that **most of the Spam messages contain numbers,** and the **majority of the Ham messages do not contain numbers.***

3.4 Data Cleaning:

3.4.1 Special character and number removal: We employ regular expressions to remove non-alphanumeric characters and numerical digits from the SMS messages, focusing solely on textual content.

3.4.2 Text normalization: The entire SMS corpus is converted to lowercase to ensure uniformity in text representation.

3.4.3 Tokenization and stopword removal: SMS messages are tokenized into individual words, and common stopwords are removed to reduce noise and enhance feature relevance.

3.4.3 Lemmatization: Words are lemmatized to transform them into their base or dictionary form, reducing feature dimensionality and standardizing word variations.

3.4.4 Corpus building: A corpus of preprocessed SMS messages is constructed, serving as input for the machine learning models.

3.5 Model Building and Evaluation:

We train multiple classification models, including Multinomial Naive Bayes, Decision Tree, Random Forest, and a Voting Ensemble, using the preprocessed and engineered dataset.

Each model is evaluated using cross-validation, with the F1-Score serving as the primary evaluation metric. This process allows us to assess the models' performance and identify the most suitable algorithm for spam detection.

Model Metrics:

The F1-Score metric is computed for each classification model: Multinomial Naive Bayes, Decision Tree, Random Forest, and the Voting Ensemble.

Precision, recall, and F1-Score are reported for both classes (ham and spam) in the classification reports.

Model Training and Evaluation:

- For each model, the training process involves fitting the model to the training set and

evaluating its performance using cross-validation.

- The classification reports provide a comprehensive overview of each model's performance, including precision, recall, and F1-Score for both classes.
- Confusion matrices are visualized to depict the true positive, false positive, true negative, and false negative predictions made by each model.

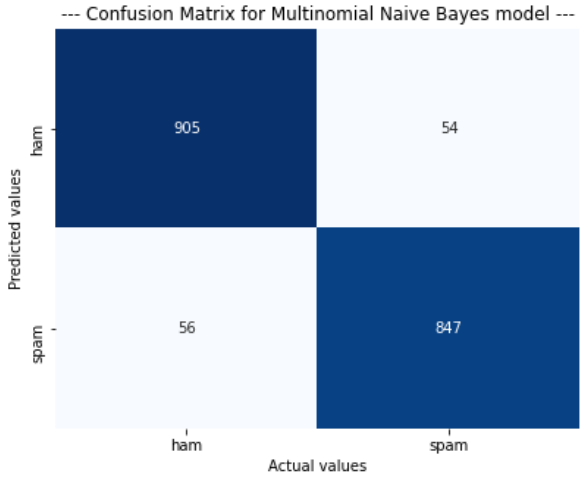


Figure 3.5.1

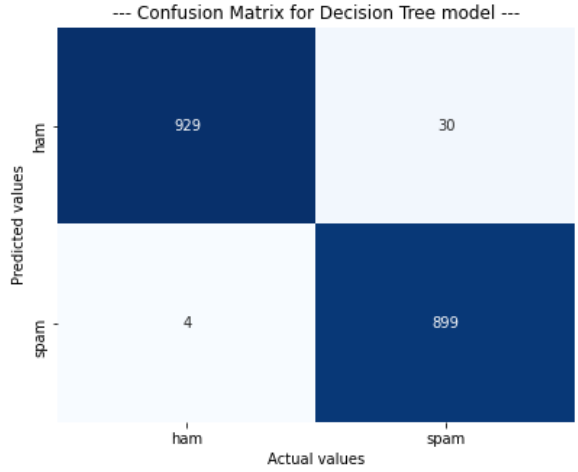


Figure:3.5.2

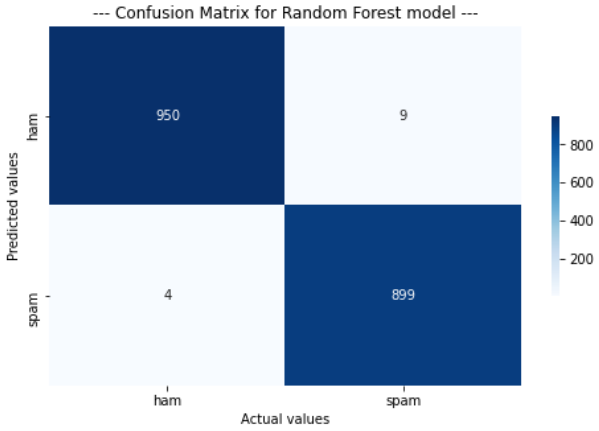


Figure 3.5.2

Model Selection and Conclusion:

- Based on the performance metrics, a conclusion is drawn regarding the most effective model for SMS spam detection.
- Random Forest emerges as the top-performing model with the highest F1-Score, followed by Decision Tree and Multinomial Naive Bayes.
- The Voting Ensemble approach, combining Decision Tree and Multinomial Naive Bayes, does not surpass the performance of Random Forest and thus is not chosen for final deployment.

4. Challenges:

1. Imbalanced Dataset: Dealing with an imbalanced dataset poses a challenge as the number of spam messages may be significantly lower than ham messages, affecting model performance and bias.
2. Feature Engineering: Selecting relevant features and engineering them effectively to improve model performance requires careful consideration and experimentation.
3. Model Selection: Choosing the most appropriate classification algorithm for SMS spam detection involves comparing multiple models and evaluating their performance metrics.
4. Optimization: Tuning hyperparameters and optimizing model parameters to enhance performance and generalization capabilities can

be time-consuming and computationally intensive.

5. Conclusion:

In conclusion, the developed SMS filtering and decluttering system demonstrates promising capabilities in automating the detection of spam messages. Through rigorous model evaluation and comparison, Random Forest emerges as the top-performing algorithm, achieving a high F1-Score of 0.994. This indicates its effectiveness in accurately classifying both spam and ham messages, thereby enhancing communication security and user experience.

6. Future Scope:

1. Enhanced Feature Engineering: Exploring additional features such as sentiment analysis, message length, and semantic analysis to improve model performance and robustness.
2. Advanced NLP Techniques: Leveraging advanced NLP techniques such as word embeddings and deep learning architectures like LSTM to capture more intricate patterns in SMS messages.
3. Ensemble Methods: Further exploring ensemble methods and model stacking techniques to combine the strengths of multiple models and improve overall prediction accuracy.
4. Real-Time Deployment: Integrating the developed system into mobile applications or communication platforms for real-time SMS spam detection and filtering.
5. Continuous Model Improvement: Implementing a feedback loop mechanism to continuously update and improve the model based on user feedback and evolving spam patterns.

7. Task Accomplishments:

Data Collection and EDA:
The responsibility of collecting the SMS dataset and conducting Exploratory Data Analysis (EDA) was assigned to Nilay Patel. Nilay successfully collected the dataset from the provided source and performed a comprehensive EDA, gaining valuable insights into the dataset's structure, composition, and distribution of spam vs. ham messages.

Feature Engineering and Data Cleaning:
Pranav Reddy Bande took charge of feature engineering and data cleaning tasks. Pranav meticulously engineered new features such as word count, currency symbol presence, and numerical digit presence from the textual SMS data. Additionally, Pranav implemented robust data cleaning techniques, including special character removal, lowercase conversion, tokenization, stopword removal, and lemmatization, ensuring the dataset was preprocessed and standardized for model training.

Model Building and Evaluation:
Pranav and Nilay collaborated on model-building and evaluation tasks. Together, we experimented with multiple classification algorithms, including Multinomial Naive Bayes, Decision Tree, Random Forest, and a Voting Ensemble. We employed cross-validation techniques to evaluate each model's performance, focusing on the F1-Score metric. Through iterative experimentation and analysis, Pranav and Nilay identified the Random Forest algorithm as the top-performing model for SMS spam detection.

Model Deployment:
The task of model deployment was a collaborative effort between Pranav and Nilay. Drawing on their collective expertise, they strategized and implemented the deployment process, ensuring seamless integration of the trained Random Forest model into the final SMS filtering and decluttering system. Their collaborative approach facilitated efficient decision-making and execution throughout the deployment phase.

Report and Documentation:
The responsibility of preparing the report and documentation was undertaken by Pranav. Your efforts ensured that all project activities, including data collection, preprocessing, model development, and deployment, were comprehensively documented. Pranav's contribution played a crucial role in summarizing the project's methodology, results, and future directions.

8. References

1. Almeida, Tiago & Gomez Hidalgo, Jose & Yamakami, Akebo. (2013). Contributions to the Study of SMS Spam Filtering: New Collection and Results.
2. Tiwari, A. K., Srivastava, A., & Srivastava, S. (2015). SMS spam filtering: Techniques and challenges. International Journal of Computer Applications, 122(18).
3. Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval. Cambridge University Press.