SMS Spam Detection Using Machine Learning

B L Sunayana
225890048
Manipal Institute of Technology
Bengaluru,India
boppana.mitblr2022@learner.manipal.edu

Ishita Singh
225890344
Manipal Institute of Technology
Bengaluru,India
ishita.mitblr2022@learner.manipal.edu

Aarushi Garg
225890388
Manipal Institute of Technology
Bengaluru,India
aarushi.mitblr2022@learner.manipal.edu

***Abstract*— This project develops an SMS Spam Detection system to filter spam messages from legitimate ones using machine learning. Various classifiers, including Logistic Regression, Naive Bayes, Decision Trees, and ensemble models like Random Forest and XGBoost, are used to classify SMS as "spam" or "ham." Text preprocessing techniques, such as tokenization, stop-word removal, and TF-IDF vectorization, convert messages into structured formats. Model performance is evaluated using accuracy, precision, recall, and F1-score. Results indicate that ensemble models, especially Random Forest and XGBoost, effectively distinguish spam, demonstrating the value of machine learning in SMS spam filtering applications.**

I. INTRODUCTION

A. Background Information
In today's digital age, SMS is a common communication channel, but it is increasingly cluttered with unsolicited, irrelevant, and potentially harmful messages known as "spam." With the high volume of SMS traffic, there is a growing need to automatically filter out these spam messages to improve user experience and security.

B. Problem Statement
This project aims to classify SMS messages as either "Spam" or "Ham" (non-spam) using various machine learning algorithms. The goal is to identify patterns in text data that distinguish spam from ham messages, thereby enabling accurate and efficient filtering.

C. Scope
The project focuses on:
Data preprocessing and feature extraction from text messages, evaluating multiple machine learning models to find the most effective for spam detection and Analyzing model performance using metrics such as accuracy, precision, recall, and F1-score.

Spam detection has been addressed using methods like Naive Bayes, Support Vector Machines, and more recent advancements in ensemble methods and boosting algorithms. This project builds upon these approaches to identify the most effective model for SMS spam classification.

Provide background information, the problem statement, and the scope of your project. Discuss the structure of the report and related work (if applicable).

II. METHODOLOGY

This project focuses on classifying SMS messages as either "spam" or "not spam." The workflow consists of data cleaning, exploratory data analysis (EDA), data preprocessing, model building, and evaluation. Several machine learning models are implemented and evaluated, including stacking and ensemble methods for improved accuracy.

A. Model Components

The model components of this SMS spam detection project begin with data cleaning and preprocessing, where the SMS dataset is loaded and prepared for analysis. Text data undergoes preprocessing steps, including converting all text to lowercase, tokenizing words, removing special characters, filtering out common stop words, and applying stemming to reduce words to their base forms, making the data more manageable for classification. Several machine learning models are then implemented using Scikit-Learn library [5] , including Logistic Regression, Support Vector Machine (SVM), Naive Bayes, Decision Tree, and Random Forest. Ensemble methods such as AdaBoost, Gradient Boosting, XGBoost, Bagging [2], and Extra Trees are also used to improve predictive accuracy and robustness. Finally, a stacking ensemble is applied, combining SVM, Naive Bayes, and Extra Trees models, with a Random Forest classifier as the final estimator, to leverage the strengths of multiple models and achieve a well-rounded classification system. This combination of preprocessing, model diversity, and ensemble techniques enhances the system's ability to accurately detect spam messages.

B. Justification of Model choice and Hyperparameters

The choice of models and hyperparameters in this SMS spam detection project was guided by the need for both accuracy and efficiency in classifying text messages. Logistic Regression was selected for its straightforward approach to binary classification, providing a solid baseline. Naive Bayes, known for its effectiveness with text data, leverages word frequency patterns and assumes feature independence, making it well-suited for handling sparse data in SMS messages. SVM was included to capture non-linear relationships with its sigmoid kernel, while Decision Tree and Random Forest models were chosen for their interpretability and ability to handle non-linear feature interactions. Ensemble methods like AdaBoost, Gradient Boosting, and XGBoost were used to boost accuracy by focusing on misclassified instances, and bagging methods [2] like Extra Trees added robustness by reducing variance. The stacking ensemble [3] further combines the strengths of SVM, Naive Bayes, and Extra Trees, using Random Forest as the meta-learner, which provides a balanced blend of model performance. Hyperparameters such as penalty='l1' in Logistic Regression, gamma=1.0 in SVM, and n_estimators=50 across the ensemble models were tuned to optimize performance while preventing overfitting, ensuring the model's effectiveness in real-world spam detection.

III. RESULTS

This project evaluates multiple machine learning models for SMS spam detection, using a labeled dataset of "spam" and "ham" (not spam) messages. Models like Logistic Regression, Naive Bayes, SVM, Decision Tree, and various ensemble methods were tested, with metrics such as accuracy and F1-score measuring performance. Ensemble and stacking models achieved the highest accuracy, highlighting the effectiveness of combining algorithms for accurate spam classification.

A. Dataset(s)

The dataset used in this project consists of labeled SMS messages, where each message is categorized as either "spam" or "ham" (not spam). This dataset provides a mix of genuine and spam messages, allowing the model to learn distinguishing features. The text data undergoes preprocessing to standardize and clean it, making it suitable for training and evaluating machine learning models on spam classification tasks.

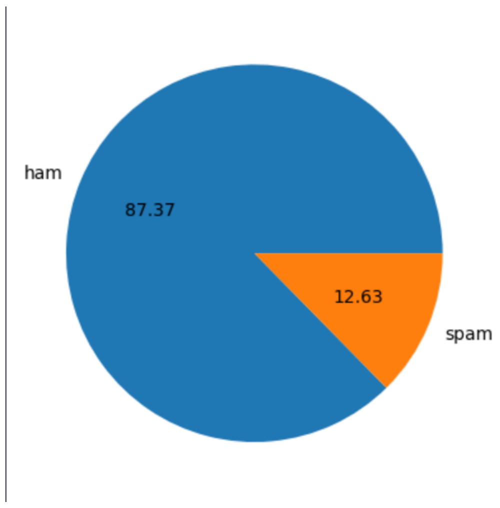

Fig 1:Pie chart of the dataset

TABLE I. TABLE OF DATASET DETAILS

S No.	Dataset description		
	Attribute	*Description*	*Value*
1	Total messages	Total SMS messages	5574
2	Spam messages	No. of Spam messages	747 (13.4%)
3	Ham messages	No. of Ham messages	4827 (86.6%)

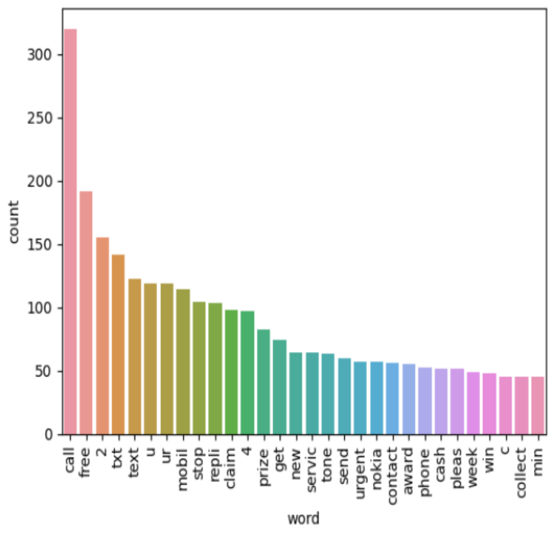

Fig 2: Bar Graph of frequently used words in Spam messages

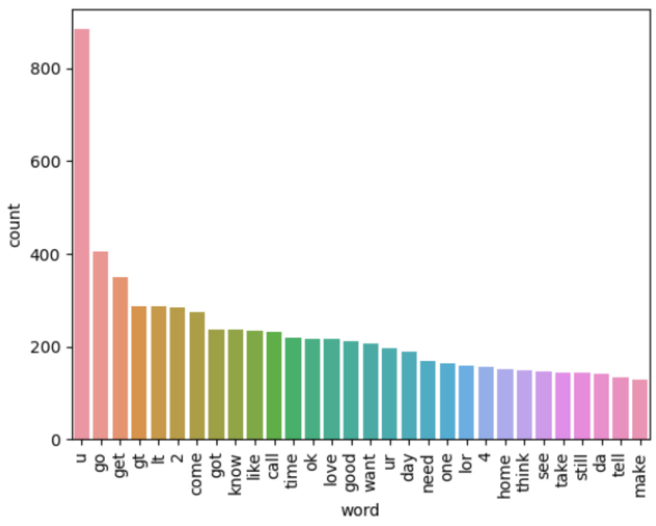

Fig 3: Bar Graph of frequently used words in Ham messages

B. Result Analysis

The results of the SMS spam detection project show that the Voting Classifier performed best, achieving an accuracy of 98.16%, precision of 99.17%. The K-Nearest Neighbors (KN) **[1]** model achieved perfect precision (100%), but with lower accuracy (90%), missing some spam messages. The Stacking Classifier also performed well with an accuracy of 97.97%, though it had lower precision. While ensemble methods (Voting and Stacking) outperformed KN, they required more computational resources.

TABLE II. TABLE OF DATASET DETAILS

Algorithm	Evaluation of Algorithms	
	Accuracy	*Precision*
KN	0.905222	1.000000
NB	0.970986	1.000000
RF	0.975822	0.982906
SVC	0.975822	0.974790
ETC	0.974855	0.974576
LR	0.958414	0.970297
AdaBoost	0.960348	0.929204
xgb	0.967118	0.926230
GBDT	0.946809	0.919192
BgC	0.958414	0.868217
DT	0.930368	0.836735

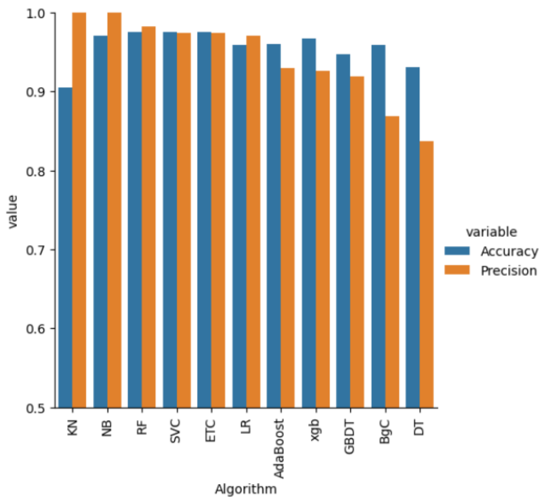

Fig 4: Bar Graph of Results

IV. DISCUSSION/ ANALYSIS

The analysis shows that the Voting Classifier delivered the best overall performance, with an accuracy of 98.16% and high precision (99.17%), indicating that it balances identifying spam messages accurately while minimizing false positives. This suggests that combining multiple models in an ensemble leads to more reliable predictions compared to individual classifiers. In contrast, K-Nearest Neighbors (KN) **[1]** achieved perfect precision (100%), meaning it correctly identified all spam messages, but its accuracy (90.52%) was lower, indicating that it missed some spam messages. The Stacking Classifier also performed well, with an accuracy of 97.97%, though its lower precision (93.98%) indicates more false positives compared to the Voting Classifier **[3]**.

When compared to traditional methods like Naive Bayes, the ensemble approaches, particularly Voting, proved superior. Naive Bayes, commonly used for text classification, often suffers from lower precision due to over-predicting spam, whereas ensemble methods effectively balance accuracy and precision. However, some limitations were encountered, including data imbalance, which could bias the models toward the majority class (non-spam), and computational complexity, as ensemble methods are more resource-intensive. Hyperparameter tuning and managing overfitting were also challenging, especially for complex models like Stacking. Future work could focus on addressing these challenges by incorporating deep learning techniques and optimizing models for real-time spam detection, as well as balancing the dataset to further improve precision.

V. CONCLUSION

This project successfully applied various machine learning models to detect spam messages, with the Voting Classifier emerging as the best performer, achieving the highest balance of accuracy and precision. The ensemble methods (Voting and Stacking) outperformed traditional models like K-Nearest Neighbors, demonstrating the value of combining classifiers for improved accuracy and robustness in text classification tasks. Key challenges included data imbalance, which affected model performance, and the computational cost of ensemble models. Future work could explore more advanced techniques like deep learning, as well as real-time deployment to improve both accuracy and efficiency. Additionally, addressing data imbalance and further optimizing model performance could enhance spam detection in real-world applications. **[4]**

REFERENCES

[1] Zhang, H., & Jin, G. (2013). KNN Text Classification Algorithm Optimization Based on Variable K Value. Journal of Software, 8(5), 1196-1201.

[2] Breiman, L. (1996). Bagging Predictors. Machine Learning, 24(2), 123-140.

[3] Rokach, L. (2010). Ensemble-based classifiers. Artificial Intelligence Review, 33, 1-39.

[4] Bhowmick, P., & Hazarika, S. M. (2012). Machine learning for email spam filtering: Review, techniques, and trends. Knowledge Engineering Review, 27(4), 407-422.

[5] Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.