AMRITA VISHWA VIDYAPEETHAM

Department of computer Science Engineering, Amritapuri

YOUTUBE SPAM COMMENT CLASSIFICATION – A REVIEW

Report

Course code & title : 22AIE213 Machine Learning

Project Group no : 02

Name	Roll number
Anuvind M P	AM.EN.U4AIE22010
Harishankar Binu Nair	AM.EN.U4AIE22023
Girish S	AM.EN.U4AIE22044
Thazhai Mugunthan G	AM.EN.U4AIE22051

I. Abstract

YouTube, as one of the largest video-sharing platforms, hosts a vast amount of user-generated content, including comments. These comments play a significant role in user engagement and community building. However, the presence of spam comments can disrupt user experience, diminish the quality of discussions, and potentially spread harmful content. Therefore, it is essential to develop effective methods to automatically detect and filter out spam comments on YouTube. The primary objective of this project is to evaluate and compare the effectiveness of various machine learning and deep learning models in classifying YouTube comments as Spam/NotSpam.

II. Introduction

YouTube, as one of the largest video-sharing platforms, receives millions of comments daily. While user engagement is beneficial, the platform is also plagued by spam comments that degrade the user experience. These spam comments often include misleading links, advertisements, and irrelevant content, disrupting meaningful interactions and potentially leading users to malicious sites. The sheer volume of comments makes it impractical for manual moderation, necessitating the development of automated spam detection systems. Effective spam detection not only enhances user experience but also maintains the integrity and safety of the platform. As the techniques employed by spammers evolve, it becomes crucial to continuously improve and adapt spam detection methods to stay ahead of these disruptions.

III. Objectives

The primary objective of this project is to develop a robust spam detection system for YouTube comments using different machine learning models. The system will preprocess the comments, perform exploratory data analysis (EDA), and evaluate multiple classifiers to determine the most effective model for spam detection.

IV. Datasets

We utilized five different datasets collected from YouTube comments on videos from various artists. The datasets include comments from videos by PSY, Katy Perry, LMFAO, Eminem and Shakira. Each dataset contains several features such as:

- *COMMENT_ID*: A unique identifier for each comment.
- *AUTHOR*: The name of the user who made the comment.
- *DATE*: The date the comment was posted.
- *CONTENT*: The actual text of the comment.
- *CLASS*: The label indicating whether the comment is spam (1) or not (0).

These datasets were merged to create a comprehensive dataset for training our models. This merged dataset enabled us to have a diverse set of comments from different contexts, improving the robustness and generalizability of our models.

The combined dataset was used to train machine learning models to classify whether a given comment is spam or not. By leveraging this diverse and rich dataset, we aimed to enhance the accuracy and performance of our spam detection model. Only the columns “CONTENT” and “CLASS” is used for training since the other columns does not provide any information to the models in this context

Another dataset contains comments from YouTube, with additional metadata including the number of likes, replies, and a spam indicator. The primary use of this dataset is to test on the models. Each entry in the dataset includes:

- *Name:* The commenter’s username, which may be useful for user behavior analysis but is not typically used in spam detection.
- *Comment*: The text of the comment, which is the primary feature for text-based models.
- *Time*: The timestamp can provide temporal context and may help in understanding trends over time.
- *Likes*: The number of likes might be an indicator of comment quality or relevance.
- *Reply Count*: The number of replies may also indicate engagement and relevance.
- *Spam*: A binary label indicating whether the comment is spam, which is the target variable for spam detection models.

The dataset contains columns for the raw comments and corresponding spam labels. Initial preprocessing was required to clean and prepare the data for analysis and modelling.

V. Data Preprocessing

A. Cleaning Process

The comments were cleaned using BeautifulSoup and Regular Expression Libraries.

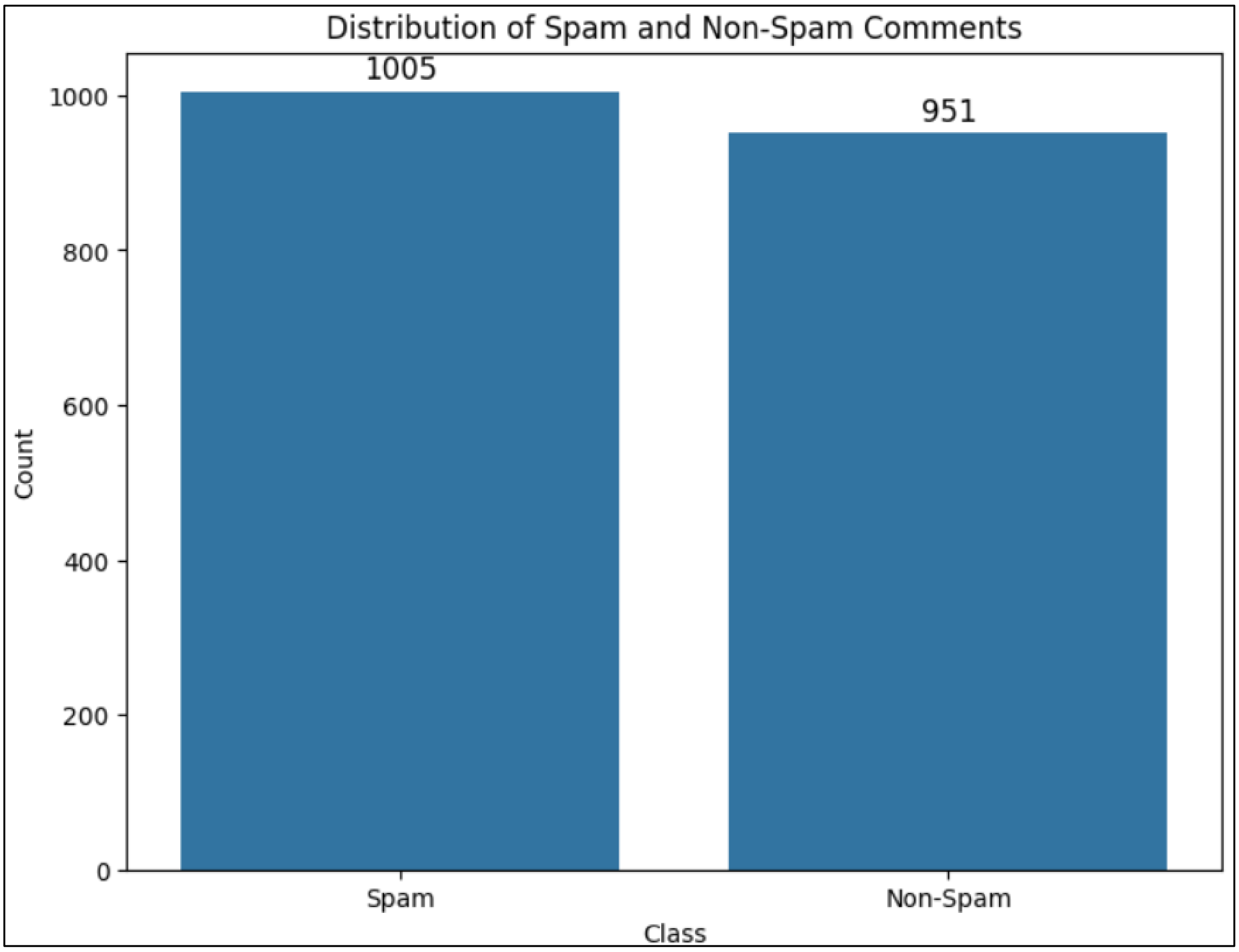
1. **HTML Tag Removal:** All HTML tags were removed using BeautifulSoup.
2. **URL Removal:** URLs were removed using regular expressions.
3. **Special Character Removal:** Non-alphanumeric characters were removed.
4. **Lowercasing:** All text was converted to lowercase.
5. **Whitespace Normalization:** Extra spaces were removed.

VI. Exploratory Data Analysis (EDA)

A. Distribution of Spam and Non-Spam Comments

A count plot was generated to show the distribution of spam and non-spam comments in the dataset. Visualizing the Distribution of data provides insight on Imbalanced Dataset. Both the classes contain almost equal number of datapoints which tells that the dataset is quite balanced.

The merged Dataset contains 1956 datapoints out of which 1005 belongs to Spam class and 951 to Non-Spam class(51.4%:48.6%).



B. Word Cloud

A word cloud provides a visual summary of the most common words in the dataset. In a word cloud, the size of each word indicates its frequency in the comments. This visualization helps quickly identify prevalent terms and phrases. For instance, in spam comments, words like "subscribe," "free," and "win" might appear prominently. Word clouds are effective for gaining a quick overview of the dataset and identifying key terms that are worth further investigation.

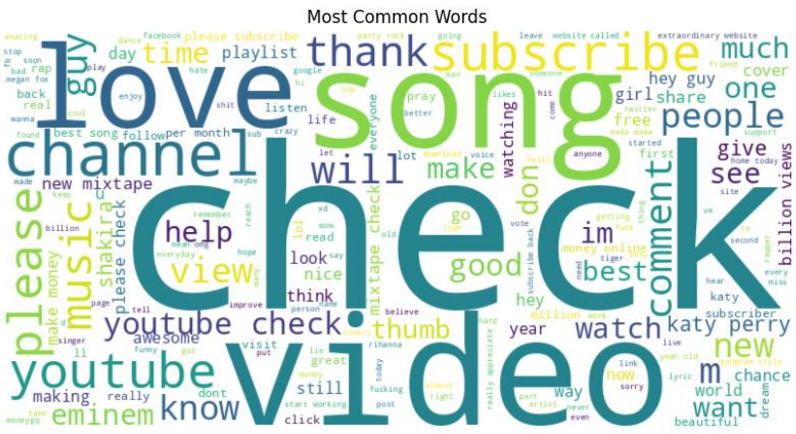
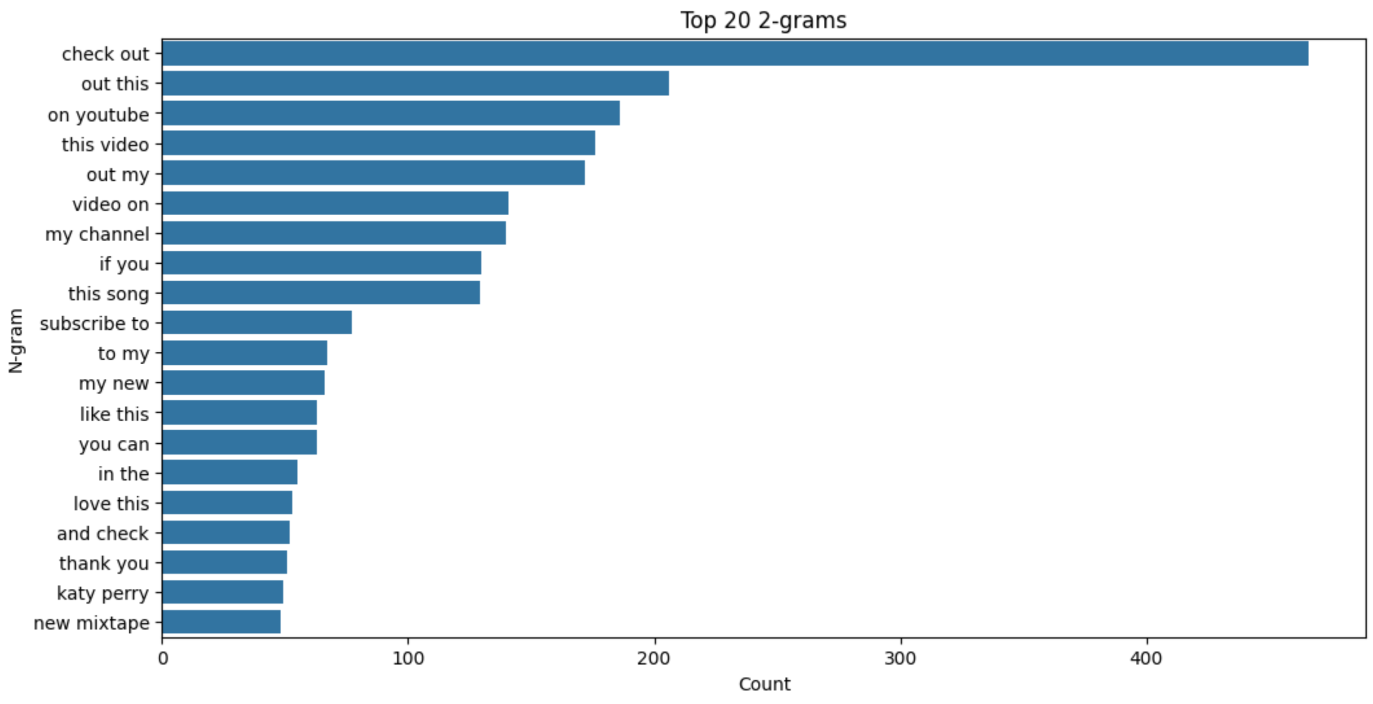
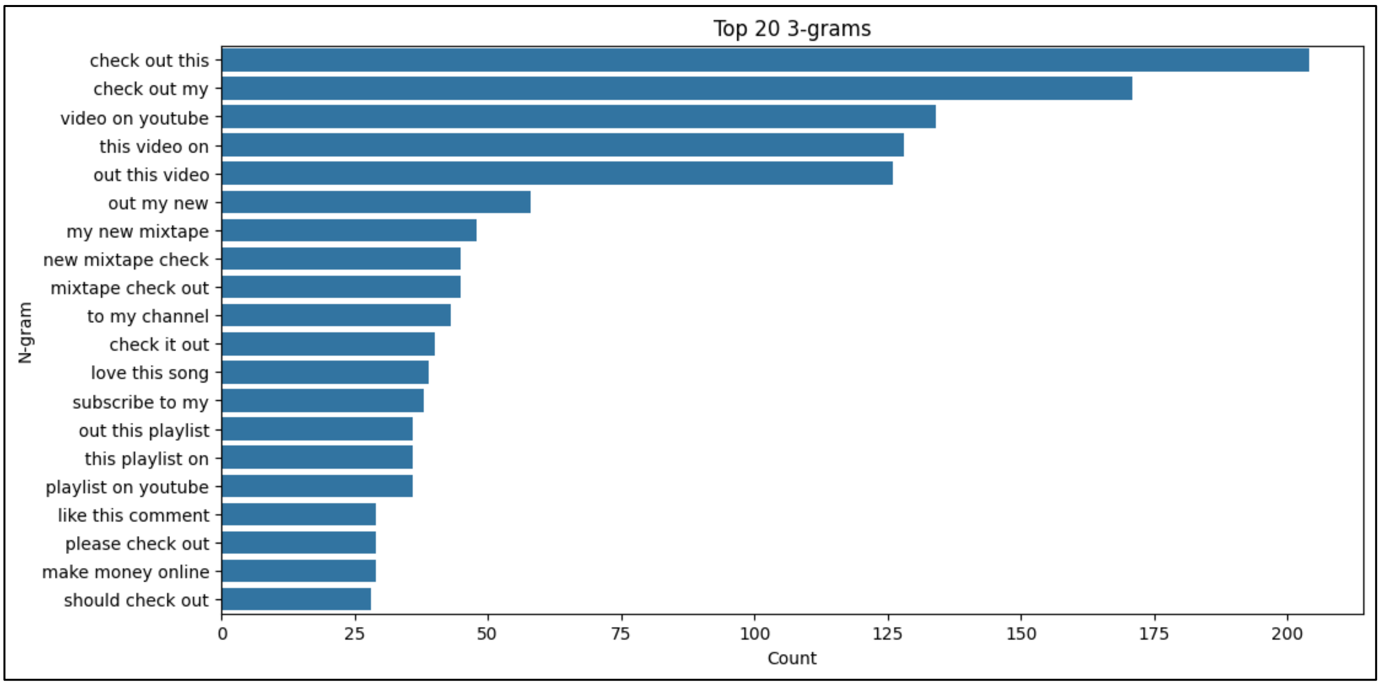


Fig: Word Cloud without Noise

C. N-gram Analysis

N-gram analysis examines contiguous sequences of words in the text. This technique is useful for identifying frequent word combinations, such as bigrams (two-word sequences) or trigrams (three-word sequences). By analysing these N-grams, we can uncover common phrases and patterns that are characteristic of spam comments. For instance, spam comments might frequently include phrases like "check out" or "click here." These insights help in feature engineering for machine learning models by providing important context and structure to the text data. Bi-grams and tri-grams were analysed to identify common phrases within the comments.





D. Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA) is a topic modelling technique used to discover underlying topics within a collection of documents. Applying LDA to the dataset helps identify common themes and subjects present in both spam and non-spam content. For example, LDA might reveal that certain spam comments are focused on promotions, while others are about giveaways or subscriptions. Understanding these topics can provide deeper insights into the nature of spam comments and aid in developing more targeted and effective spam detection models.

VII. Model Development

The data was split into training and testing sets using a 70-30 split. The data was fed into a sklearn pipeline with TFIDF feature extractor and passed to the model. The Pipeline was then given for Hyperparameter Tuning using sklearn' s GridSearchCV with a 5-fold cross validation.

A. Feature Extraction

TF-IDF (Term Frequency-Inverse Document Frequency)

TF-IDF is a statistical measure used to evaluate the importance of a word in a document relative to a collection of documents (corpus). It is a common technique for transforming text into a numerical representation for machine learning algorithms. The TF-IDF value increases proportionally with the number of times a word appears in the document but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general.

Advantages of TF-IDF include its ability to filter out common words that are less informative (such as "the", "and", "is") and its effectiveness in handling large vocabularies. By representing text data in a way that reflects the importance of words in context, TF-IDF aids in improving the performance of various text-based machine learning models. Common uses of TF-IDF include information retrieval, text mining, and natural language processing tasks such as document classification and sentiment analysis.

In this project, TF-IDF was used to transform the raw comment text into numerical features. This transformation allowed the models to learn patterns from the text data effectively, improving the classification accuracy for detecting spam comments. By considering both the term frequency and the inverse document frequency, TF-IDF provided a balanced representation of word importance across the dataset.

B. Hyperparameter Tuning

We implemented hyperparameter tuning using GridSearchCV to optimize the performance of our models: SVM, KNN, Multinomial Naive Bayes, Random Forest, and Logistic Regression. Hyperparameter tuning is a crucial step in model optimization as it helps in identifying the best combination of parameters that lead to the highest predictive accuracy.

GridSearchCV systematically works through multiple combinations of parameter values, cross-validating each combination to determine the best parameters. For each model, we defined a grid of potential hyperparameters. For instance, for the SVM, we might have varied the kernel type, regularization parameter (C), and gamma. For KNN, the number of neighbors (k) and the distance metric could be tuned. For Multinomial Naive Bayes, the smoothing parameter (alpha) would be considered. Random Forest might involve tuning the number of trees, maximum depth, and minimum samples split. For Logistic Regression, we could adjust the regularization strength (C) and solver.

Using GridSearchCV, each model is trained multiple times with different combinations of these parameters, and the performance is validated using cross-validation. This approach helps in avoiding overfitting, as the model is evaluated on different subsets of the training data.

The result of this process is the identification of the best set of hyperparameters for each model. These optimal parameters are then used to train the final model, ensuring that we achieve the highest possible accuracy on the test data. Hyperparameter tuning, therefore, enhances the model's performance by fine-tuning the learning process, ensuring robust and reliable predictions in the spam classification task.

C. Models Evaluated

1. **Multinomial Naive Bayes (NB)**
2. **Logistic Regression (LR)**
3. **Random Forest Classifier (RF)**
4. **K-Nearest Neighbors (KNN)**
5. **Support Vector Machine (SVM)**

1. Multinomial Naive Bayes (NB)

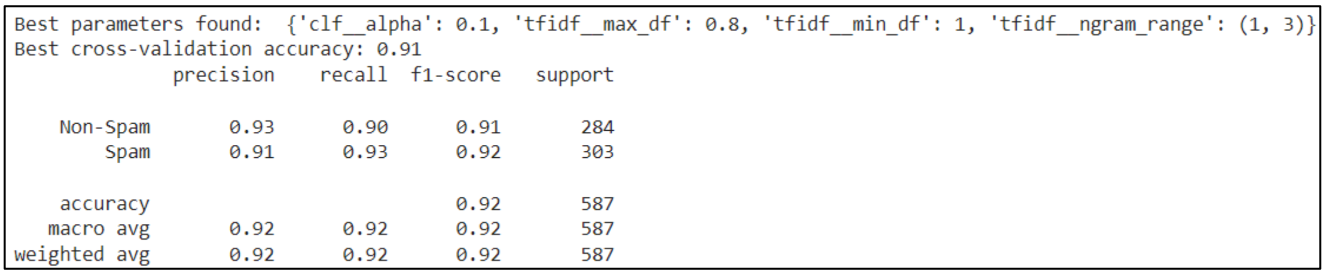
Multinomial Naive Bayes (MNB) is a probabilistic learning algorithm commonly used for text classification tasks. It is based on Bayes' theorem and works particularly well with word frequency features. MNB is effective for spam detection because it can handle large vocabularies and the independence assumption (that the presence of a particular feature in a class is unrelated to the presence of any other feature) works reasonably well in practice for text data. Advantages of MNB include its efficiency in terms of both time and space, its performance with a large number of features, and its simplicity in implementation and interpretation. MNB is commonly used in email spam detection, document categorization, and sentiment analysis.

Tuned Parameters:

- alpha: 1.0
- fit_prior: True

Multinomial Naive Bayes achieved a commendable 91% accuracy, leveraging the probabilistic nature of the model. This model's performance is particularly notable given its assumption of feature independence, which often holds true for text data.

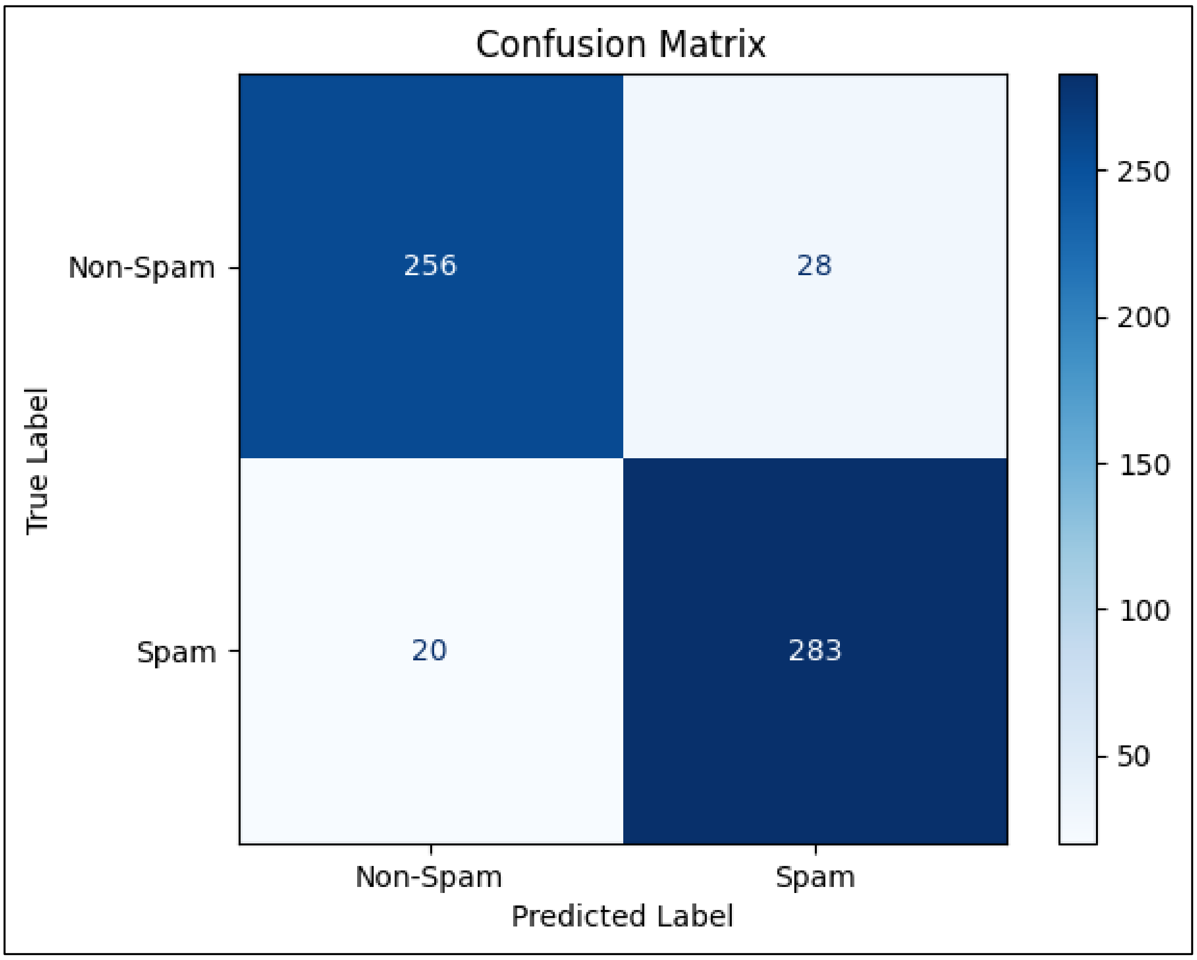
Results: The Multinomial Naive Bayes model achieved an accuracy of 92%, with precision, recall, and F1-score values of 92%, 92%, and 92% respectively. The confusion matrix showed 283 true positives, 256 true negatives, 28 false positives, and 20 false negatives.4



```
Best parameters found:  {'clf__alpha': 0.1, 'tfidf__max_df': 0.8, 'tfidf__min_df': 1, 'tfidf__ngram_range': (1, 3)}
Best cross-validation accuracy: 0.91
              precision    recall  f1-score   support

    Non-Spam       0.93      0.90      0.91       284
        Spam       0.91      0.93      0.92       303

    accuracy                           0.92       587
   macro avg       0.92      0.92      0.92       587
weighted avg       0.92      0.92      0.92       587
```



2. Logistic Regression

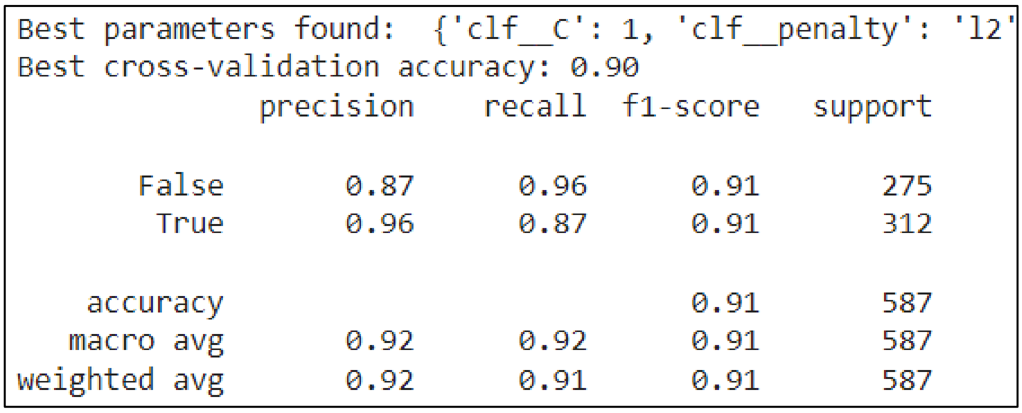
Logistic Regression is a statistical model used for binary classification problems. It predicts the probability that a given input belongs to a certain class. In the context of spam detection, Logistic Regression can provide probabilities for each comment being spam or non-spam, which allows for setting flexible thresholds. Advantages of Logistic Regression include its ease of implementation and understanding, its ability to output probability scores, and its performance when the relationship between the features and the outcome is linear. Common uses of Logistic Regression include credit scoring, disease prediction and diagnosis, and spam and fraud detection.

Tuned Parameters:

- Penalty : 'l2'
- C : 1.0
- solver: 'lbfgs'

Logistic Regression performed well with a 90% accuracy, benefiting from its simplicity and efficiency. Its ability to handle large feature spaces made it a solid choice for this classification task.

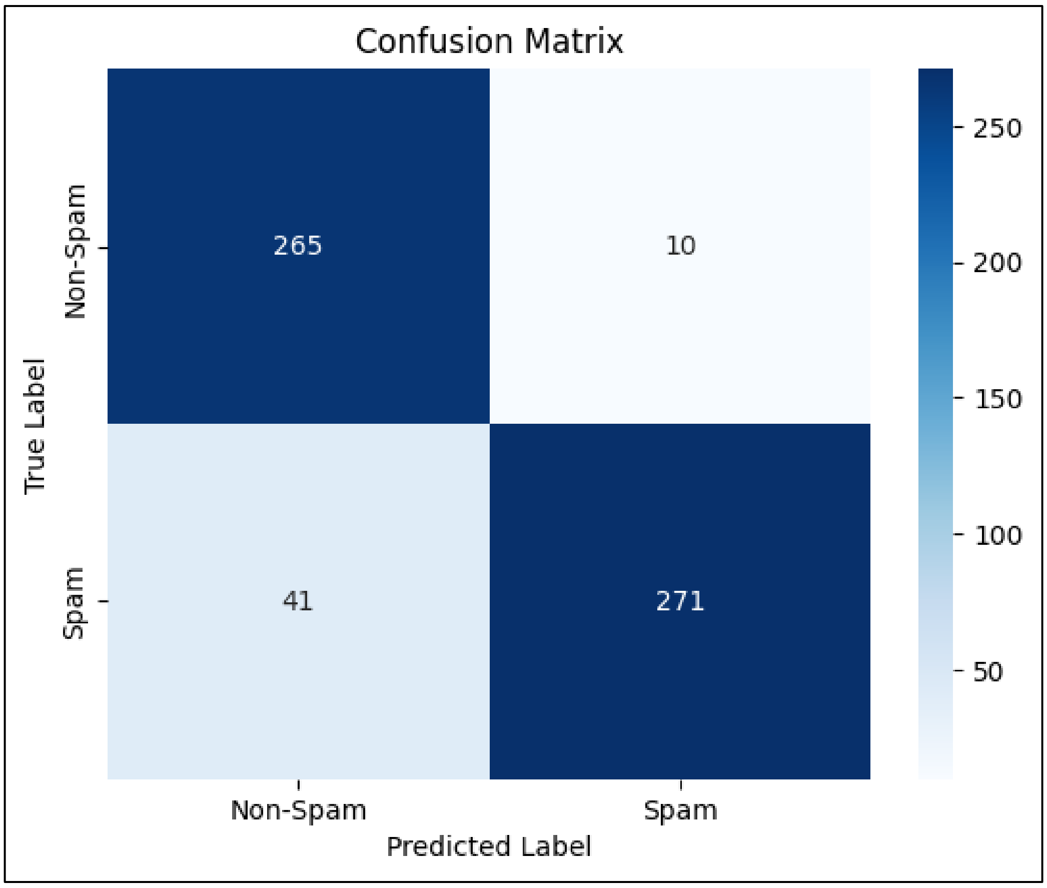
Results: The Logistic Regression model achieved an accuracy of 91%, with precision, recall, and F1-score values of 92%, 91%, and 91% respectively. The confusion matrix showed 271 true positives, 265 true negatives, 10 false positives, and 41 false negatives.



```
Best parameters found:  {'clf__C': 1, 'clf__penalty': 'l2'
Best cross-validation accuracy: 0.90
              precision    recall  f1-score   support

       False       0.87      0.96      0.91       275
        True       0.96      0.87      0.91       312

    accuracy                           0.91       587
   macro avg       0.92      0.92      0.91       587
weighted avg       0.92      0.91      0.91       587
```



3. Random Forest Classifier

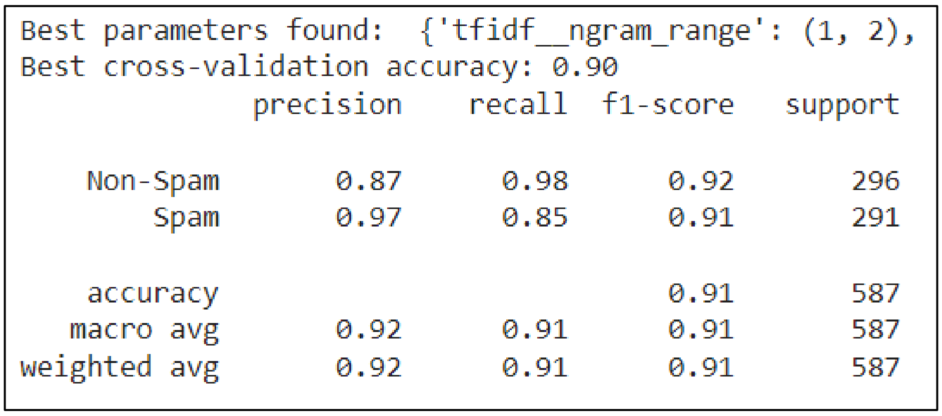
Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes (classification) of the individual trees. It is robust to overfitting and can handle large datasets with higher dimensionality. Advantages of Random Forest include its high accuracy, its ability to handle large datasets, and its reduction of overfitting by averaging multiple decision trees. Common uses of Random Forest include classification tasks in finance, healthcare, and marketing, feature selection, and anomaly detection.

Tuned Parameters:

- n_estimators : 200
- max_features : 'auto'
- max_depth : 90
- min_samples_split : 2
- min_samples_leaf : 1
- bootstrap : True

The Random Forest model displayed robust performance with a 90% cross-validation accuracy. Its ensemble approach, leveraging multiple decision trees, contributed to its stability and reliability in spam detection.

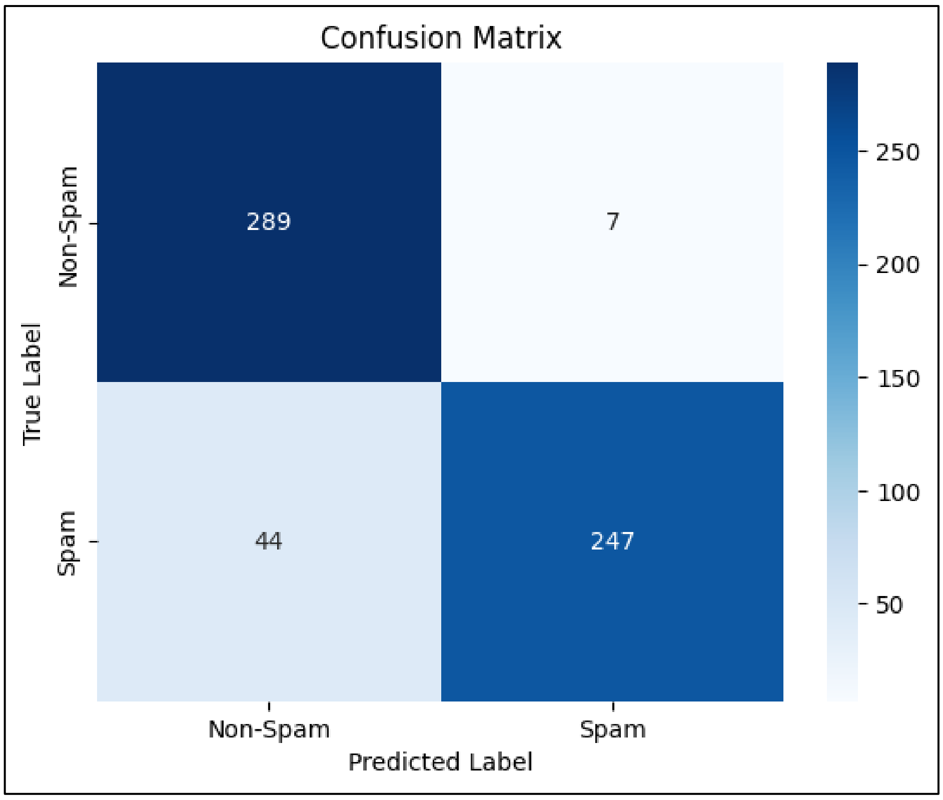
Results: The Random Forest model achieved an accuracy of 91%, with precision, recall, and F1-score values of 92%, 91%, and 91% respectively. The confusion matrix showed 247 true positives, 289 true negatives, 7 false positives, and 44 false negatives.



```
Best parameters found:  {'tfidf__ngram_range': (1, 2),
Best cross-validation accuracy: 0.90
              precision    recall  f1-score   support

    Non-Spam       0.87      0.98      0.92       296
        Spam       0.97      0.85      0.91       291

    accuracy                           0.91       587
   macro avg       0.92      0.91      0.91       587
weighted avg       0.92      0.91      0.91       587
```



4. K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a type of algorithm used for classification and regression tasks in machine learning. Unlike parametric models, KNN is non-parametric and doesn't make assumptions about the underlying data distribution. It classifies a new data point based on the majority class of its k nearest neighbors in the feature space. This approach makes KNN straightforward to implement and effective for small datasets with minimal noise.

One notable advantage of KNN is its simplicity, which makes it easy to understand and apply. Moreover, it can handle complex decision boundaries and doesn't require training time since it stores all training data points for classification. This makes it particularly useful when the data is well-labeled and the task involves finding similar items or patterns, such as in recommender systems, image recognition, and video classification.

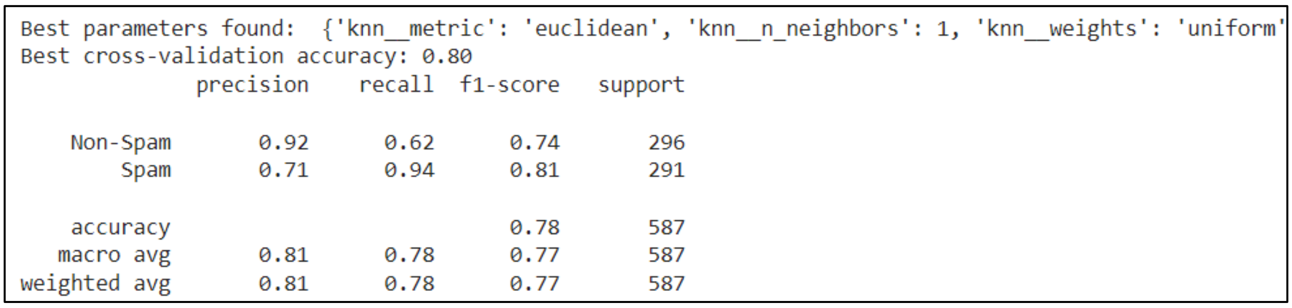
In essence, KNN leverages the proximity of data points in a high-dimensional space to make predictions, making it a versatile tool in various machine learning applications

Tuned Parameters:

- n_neighbors: 5
- weights: 'uniform'
- algorithm: 'auto'

The KNN model, while achieving a lower accuracy of 80%, provides insights into how proximity-based methods can be applied to text classification. Its simplicity in implementation makes it a useful baseline model despite its comparatively lower performance.

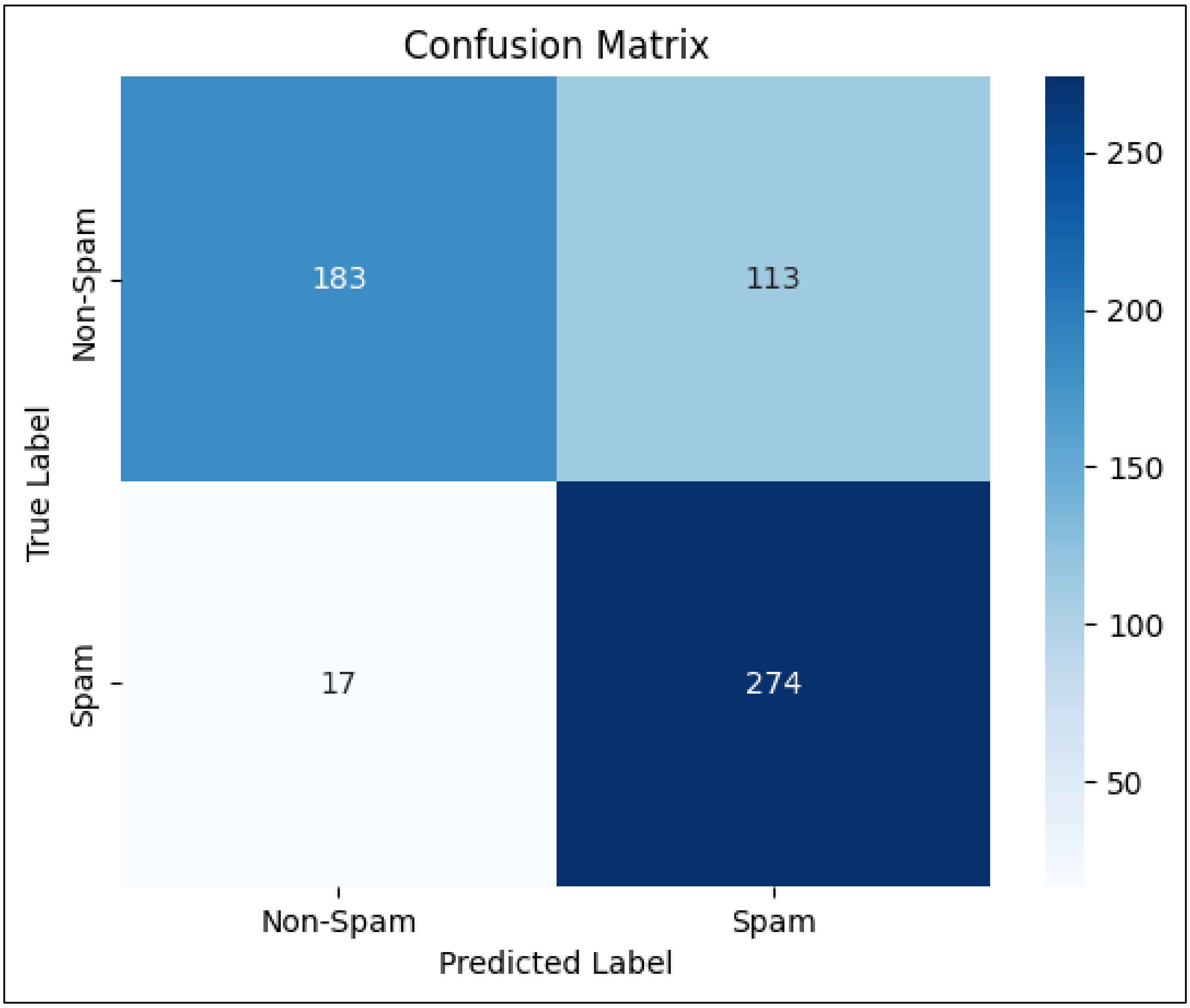
Results: The K-Nearest Neighbors model achieved an accuracy of 78%, with precision, recall, and F1-score values of 81%, 78%, and 77% respectively. The confusion matrix showed 274 true positives, 183 true negatives, 113 false positives, and 17 false negatives.



```
Best parameters found:  {'knn__metric': 'euclidean', 'knn__n_neighbors': 1, 'knn__weights': 'uniform'
Best cross-validation accuracy: 0.80
              precision    recall  f1-score   support

    Non-Spam       0.92      0.62      0.74       296
        Spam       0.71      0.94      0.81       291

    accuracy                           0.78       587
   macro avg       0.81      0.78      0.77       587
weighted avg       0.81      0.78      0.77       587
```



5. Support Vector Machine (SVM)

Support Vector Machine is a powerful, supervised machine learning algorithm used for classification or regression challenges. It works well for both linear and non-linear data by using a kernel trick to transform the data and find an optimal boundary between the possible outputs. Advantages of SVM include its effectiveness in high-dimensional spaces, robustness to overfitting (especially in high-dimensional datasets), and its ability to model non-linear decision boundaries with the use of kernel functions. Common uses of SVM include image and text classification, bioinformatics (gene and protein classification), and handwriting recognition.

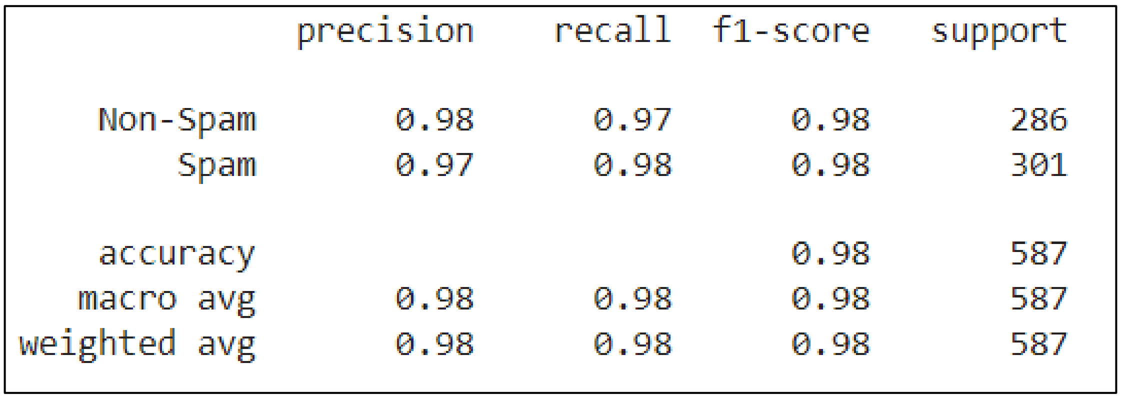
Tuned Parameters:

- svc__C: 1
- svc__gamma: 'scale'
- svc__kernel: ‘rbf’

Cross-Validation Accuracy: 98%

The SVM model achieved the highest cross-validation accuracy, indicating its strong capability in distinguishing between spam and non-spam comments. Its performance is attributed to the precise tuning of hyperparameters, which maximized its effectiveness.

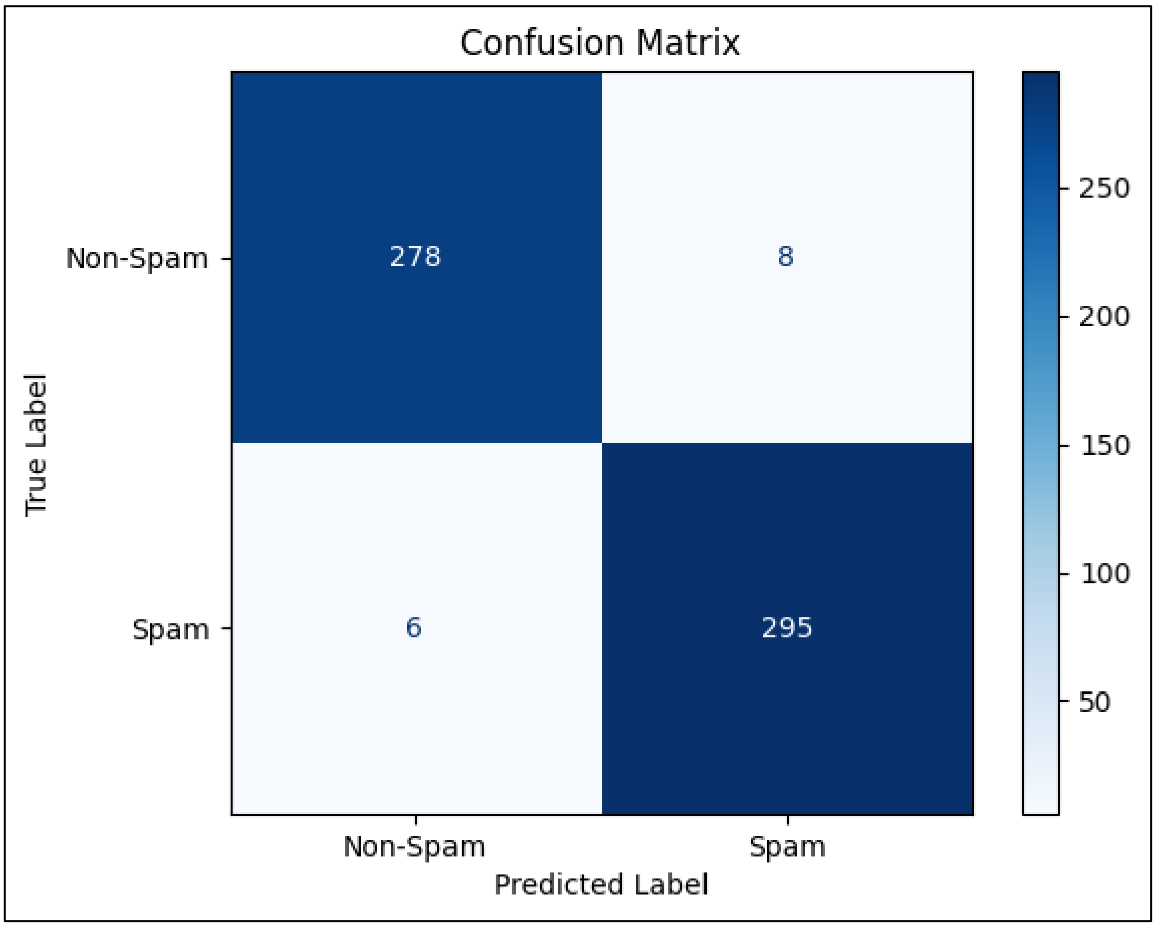
Results: The Support Vector Machine model achieved an accuracy of 98%, with precision, recall, and F1-score values of 98%, 98%, and 98% respectively. The confusion matrix showed 295 true positives, 278 true negatives, 8 false positives, and 6 false negatives.



```
              precision    recall  f1-score   support

    Non-Spam       0.98      0.97      0.98       286
        Spam       0.97      0.98      0.98       301

    accuracy                           0.98       587
   macro avg       0.98      0.98      0.98       587
weighted avg       0.98      0.98      0.98       587
```



VIII. Future Scope

Future work could focus on further fine-tuning and exploring additional models to enhance spam detection accuracy. One potential direction is to experiment with different feature extraction techniques or leverage more advanced models to capture complex patterns in the data. Additionally, exploring ensemble methods that combine multiple models could lead to improved performance by leveraging the strengths of each model.

To classify bot comments more effectively, it is essential to consider additional features beyond the text content. Incorporating features such as profile pictures (pfp), previous interactions, and activity patterns can provide valuable insights into the behavior and characteristics of users. For instance, analyzing the consistency and authenticity of profile pictures can help identify bots that use generic or stolen images. Examining previous interactions, such as comment history and engagement patterns, can reveal suspicious behavior indicative of automated accounts. Furthermore, analyzing activity patterns, such as the frequency and timing of comments, can help distinguish between genuine users and bots that operate on predefined schedules.

Integrating these enhanced models into a real-time system could provide practical benefits for YouTube content moderation. By continuously monitoring and analyzing comments as they are posted, the system can quickly identify and flag spam or bot-generated content. This proactive approach can help maintain a cleaner and more user-friendly platform, improving the overall user experience and fostering a healthier online community. Additionally, the insights gained from analyzing spam and bot comments can be used to refine and adapt the moderation strategies over time, ensuring that the system remains effective against evolving spam tactics and bot behavior.

IX. Conclusion

This project aimed to detect and classify spam comments on YouTube using various machine learning models. The steps undertaken included data preprocessing, exploratory data analysis, model training, and evaluation. The dataset comprised both spam and non-spam comments, which were analyzed to understand their distribution. Word count distribution showed varying lengths of comments, and a word cloud revealed the most common words used. N-gram analysis identified frequent word combinations, while sentiment analysis demonstrated the distribution of sentiment scores within the comments.

Different models were employed for classification, including MultiNomial Naive Bayes, Logistic Regression, and Random Forest. Each model was trained and tested on the dataset, with performance evaluated through confusion matrices and classification reports. The results showed that while each model had its strengths, the performance metrics varied.

Ultimately, an SVM (Support Vector Machine) with Radial Basis Kernel model was implemented, which achieved an impressive 98% accuracy. This high level of accuracy indicates the model's effectiveness in correctly classifying spam and non-spam comments. Given these results, the SVM model was selected as the best model for this task.