# A

# Project Report

# on

# Spam Detection using NLP

# CSCI 642 Natural Language Processing

## Under the guidance of

Dr. Maninder Singh


## Department of Computer Science

## St. Cloud State University


## Submitted by

## Team #2


Deepika Kakkera

Divya Darsi

Jyoshika Sripathi

Karthik Kommera

# Table of Contents

# 1. TEAM MEMBERS CONTRIBUTIONS

| Name | Contributions |
|---|---|
| Divya Darsi | Introduction,<br>Proposed Solution,<br>Research Question,<br>Programming/Execution,<br>Hardware/software descriptions,<br>Experimental Design,<br>Experimental procedure,<br>Result and Discussion,<br>Conclusion,<br>Documentation & Presentation. |
| Deepika Kakkera | Problem Statement,<br>Motivation,<br>Data Set,<br>Preprocessing Data,<br>Experimental Design,<br>Experimental procedure,<br>Result and Discussion,<br>Conclusion,<br>Documentation & Presentation. |
| Jyoshika Sripathi | Proposed Solution,<br>Major Functionalities,<br>Research Question ,<br>Experiment Design,<br>Describe pre-processing of your dataset,<br>Experimental Design,<br>Experimental procedure,<br>Result and Discussion,<br>Conclusion,<br>Documentation & Presentation. |
| Karthik Kommera | Background,<br>Project Risks,<br>Data Set,<br>Experiment Validation,<br>Approach refinement,<br>Experimental Design,<br>Experimental procedure,<br>Result and Discussion,<br>Conclusion,<br>Documentation & Presentation. |

# 2. INTRODUCTION

Messaging has revolutionized communication, offering fast and reliable connectivity across vast distances. Yet, its widespread adoption has made messaging platforms a target for spam – unwanted communications disrupting our digital experience. Spam takes many forms, from ads to scams, wasting time and posing risks like malware. Spammers employ misleading information, exploiting trust and human vulnerabilities with phishing and social engineering. This softens the line between legitimate and fraudulent messages, challenging users to discern between them. Thus, vigilance and awareness are crucial in navigating the digital landscape safely.

Spam, originating from compromised computers, floods our inboxes, creating a major challenge for digital communication. Its sheer volume overwhelms email servers and messaging platforms, causing delays, and eroding user confidence. Despite efforts to combat it with blacklists and filters, spam detection struggles to keep up with evolving tactics. Spammers use sophisticated methods like obfuscation and distributed botnets to evade detection, taking advantage of human vulnerabilities. This constant barrage disrupts normal communication and drains valuable resources.

As internet usage and email marketing grow, spam continues to proliferate, luring recipients with promises of quick wealth or questionable products. Consequently, the need for effective spam detection solutions has never been more critical. By harnessing advanced technologies such as machine learning and behavioural analysis, our project aims to develop robust algorithms capable of accurately identifying and filtering out spam across various messaging platforms. By improving the security and reliability of digital communication, we aim to protect users and uphold the integrity of online interactions in an increasingly connected world.

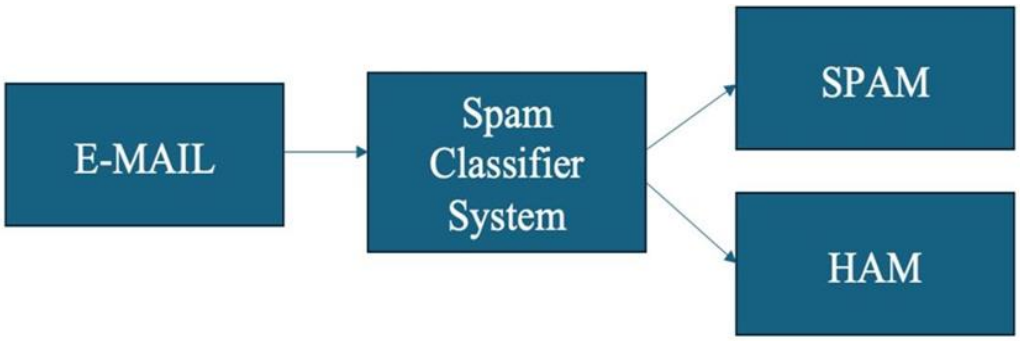



**Figure 2.1: Spam Classifier**

# 3.PROBLEM STATEMENT

## 3.1 Introduction to the Surge of Spam Messages

The widespread adoption of mobile devices and digital communication platforms has led to a dramatic increase in both the quantity and complexity of spam messages. As illustrated in Figure 1, the rapid expansion of internet users and online communication channels has provided spammers with ample opportunities to exploit.

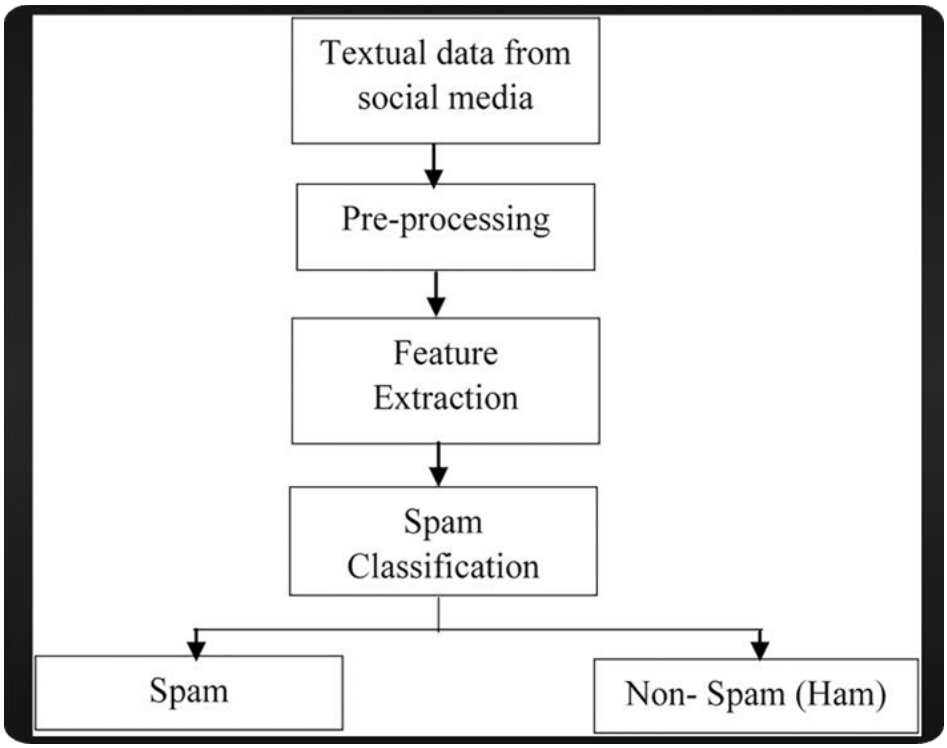



**Figure 3.1: Block diagram illustrating the flow of spam messages into users' inboxes.**

## 3.2 Multifaceted Risks and Inconveniences for Users

The influx of spam messages presents various risks and inconveniences for both individuals and organizations. Studies, as depicted in Figure 3.2 highlight that the average user encounters a substantial volume of spam messages daily, raising the risk of falling for fraudulent schemes. Moreover, the repercussions of succumbing to spam-related scams can be dire. Cases of financial loss and identity theft stemming from phishing attempts emphasize the critical importance of implementing effective spam detection and mitigation strategies.

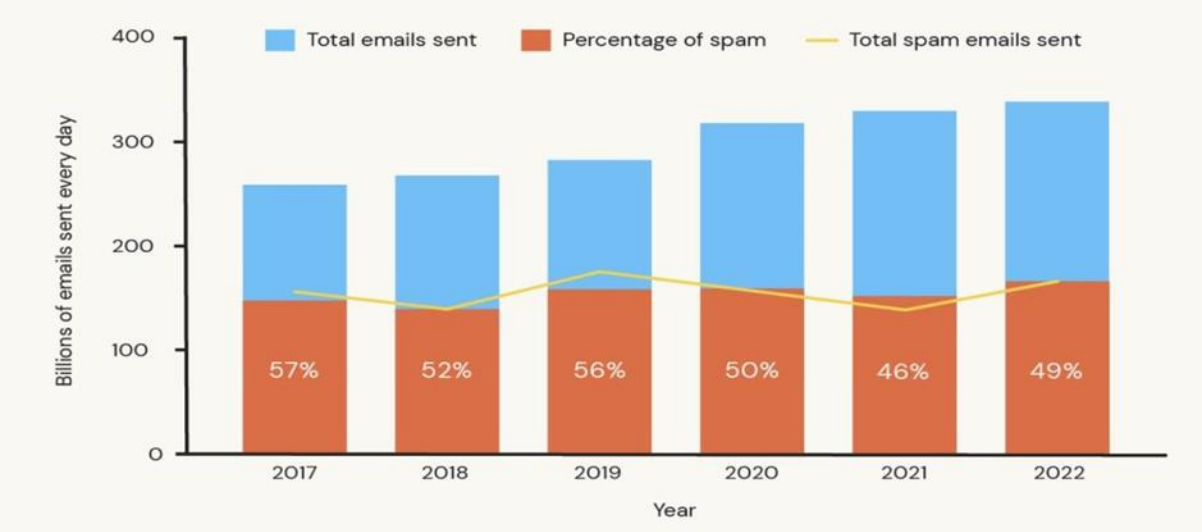



**Figure 3.2: Bar graph depicting the average number of spam messages received by users daily.**

## 3.3 Disruption to User Experience and Productivity

The incessant influx of spam messages disrupts users' experience and productivity. The clutter of unwanted messages impedes efficient communication and distracts users from important tasks, resulting in decreased productivity and increased frustration.

## 3.4 Need for Effective Spam Detection and Mitigation Strategies

Given the pervasive nature of the spam problem, there is an urgent need for effective detection and mitigation strategies tailored to mobile messaging platforms. By leveraging advanced technologies such as machine learning algorithms, natural language processing techniques, and behavioural analysis, users can be empowered with robust tools for identifying and filtering out spam messages in real-time.

# 4. MOTIVATION

Spam detection plays a crucial role in modern communication systems due to its far-reaching impacts on various aspects of digital communication. The following are key reasons why effective spam detection is essential:

**Enhancing User Experience:** Spam inundates users' inboxes, cluttering their communication channels with unwanted messages. This deluge of irrelevant content not only hampers users' ability to find important messages but also leads to frustration and dissatisfaction. Effective spam detection helps streamline users' communication experiences by filtering out unwanted messages and ensuring that their inboxes remain clutter-free.

**Protecting Privacy and Security:** Spam messages often contain malicious content such as phishing links, malware, or fraudulent schemes. Clicking on these links or downloading attachments can compromise users' personal information, financial data, and even their devices. By detecting and blocking spam messages, communication systems can mitigate the risk of security breaches, identity theft, and other cyber threats, thus safeguarding users' privacy and security.

**Preventing Financial Loss:** Phishing scams and fraudulent schemes embedded within spam messages can lead to significant financial losses for individuals and organizations. These scams often trick users into divulging sensitive information, such as login credentials or banking details, which can then be used for fraudulent purposes. Effective spam detection helps thwart these scams by identifying and flagging suspicious messages before they can cause financial harm.

**Maintaining System Reliability:** The unchecked proliferation of spam can strain communication systems, leading to performance issues, network congestion, and service disruptions. Inefficient spam filtering mechanisms may inadvertently block legitimate messages or allow malicious content to bypass security measures, undermining the reliability and trustworthiness of the entire communication infrastructure. Robust spam detection algorithms help maintain system reliability by accurately identifying and handling spam messages, ensuring smooth and uninterrupted communication for users.

**Preserving Reputation and Trust:** A high volume of spam can tarnish the reputation of communication platforms and service providers. Users may lose trust in platforms that fail to effectively manage spam, leading to decreased user engagement, churn, and reputational damage. By implementing effective spam detection measures, communication systems demonstrate their commitment to user safety and satisfaction, thereby preserving trust and credibility among users and stakeholders.

# 5. BACKGROUND

Spam detection techniques have evolved significantly over the years, ranging from simple rule-based methods to sophisticated machine learning approaches and hybrid models. Each approach has its strengths and weaknesses, and the effectiveness of spam detection often depends on the context and characteristics of the communication platform.

**Rule-Based Methods:** Rule-based spam detection relies on predefined sets of rules or patterns to identify spam messages. These rules may include criteria such as keyword frequency, message length, sender reputation, and presence of specific content or formatting. While rule-based methods are relatively simple to implement and computationally efficient, they may struggle to adapt to new spam tactics and may generate false positives or false negatives.

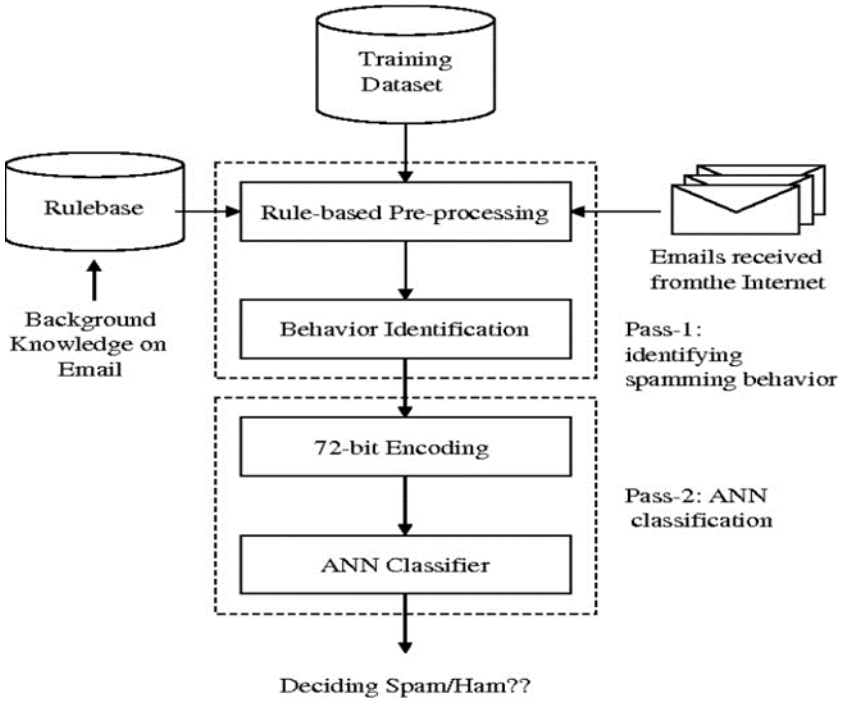



**Figure 5.1: Behaviour-based spam detection using a hybrid method of rule-based techniques and neural networks.**

**Machine Learning Approaches:** Machine learning techniques have gained popularity in spam detection due to their ability to automatically learn and adapt from data. Supervised learning algorithms, such as Support Vector Machines (SVM), Naive Bayes, and Decision Trees, are trained on labelled datasets of spam and non-spam messages to identify patterns and features indicative of spam. Unsupervised learning algorithms, such as clustering and anomaly detection, can also be used to detect spam without labelled data. Machine learning approaches

offer the advantage of scalability and adaptability, but they require large amounts of labelled data for training and may be susceptible to adversarial attacks.

**Hybrid Models:** Hybrid spam detection models combine multiple techniques, such as rule-based methods and machine learning algorithms, to improve detection accuracy and robustness. For example, a hybrid model may use rule-based filtering to flag obvious spam messages and then use machine learning algorithms to classify borderline cases. By leveraging the strengths of both approaches, hybrid models can achieve better performance and reduce false positives/negatives compared to individual techniques alone.

**Notable Research and Evolution of Spam Detection Technology:** Over the years, researchers have developed innovative approaches to spam detection, continually improving the accuracy and efficiency of detection systems. Some notable research contributions and advancements in spam detection technology include:

**Content-Based Filtering**: Early spam detection systems relied heavily on content-based filtering, where messages were analysed based on their textual content, including keywords, phrases, and structural features. Researchers explored various algorithms and techniques to improve the effectiveness of content-based filtering, such as Bayesian filtering and regular expression matching.

**Sender Reputation Analysis:** Another significant advancement in spam detection is the integration of sender reputation analysis, where the reputation of email senders is assessed based on their past behaviour and characteristics of their messages. Sender reputation plays a crucial role in determining the likelihood of a message being spam, helping to reduce false positives and improve detection accuracy.

**Behavioural Analysis:** Recent research has focused on incorporating behavioural analysis techniques into spam detection systems. Behavioural analysis considers users' interactions with messages and communication patterns to identify anomalies and suspicious activities indicative of spam. Techniques such as clickstream analysis, user profiling, and network traffic analysis have shown promise in detecting sophisticated spam tactics, such as spear-phishing and social engineering attacks.

**Adversarial Machine Learning:** With the increasing sophistication of spam tactics, researchers are exploring techniques to make machine learning-based spam detection systems more robust against adversarial attacks. Adversarial machine learning involves training models to withstand attacks designed to deceive or manipulate them, such as adversarial examples and evasion attacks. By improving the resilience of machine learning models, researchers aim to enhance the reliability and effectiveness of spam detection systems in real-world scenarios.

# 6. PROPOSED SOLUTION

**Proposed Solution:** Enhancing Email Spam Detection with Multinomial Naive Bayes

**Research Question:** *Which feature extraction method, TF-IDF or Bag-of-Words (BOW), yields higher accuracy in classifying emails as spam or ham when employed with a Multinomial Naive Bayes algorithm?*

**Introduction:** Email spam detection is pivotal for maintaining the security and efficiency of communication systems. With the reliance on machine learning algorithms, traditional methods categorize emails into spam or ham (non-spam) categories. In this project, our objective is to bolster the robustness of email spam detection systems by integrating Multinomial Naive Bayes (MNB) with two distinct feature extraction methods: Bag-of-Words (BOW) and TF-IDF.

## Approach:

**Data Preprocessing:** We initiated the process by subjecting textual data to comprehensive preprocessing steps. These steps included tokenization, stop word removal, lemmatization, stemming, and part-of-speech tagging. The aim was to render the text data suitable for subsequent feature extraction.

**Feature Extraction:** Two feature extraction techniques were employed.

- Bag-of-Words (BOW): This method represents documents as word frequency vectors, capturing the occurrence of each word within the document.

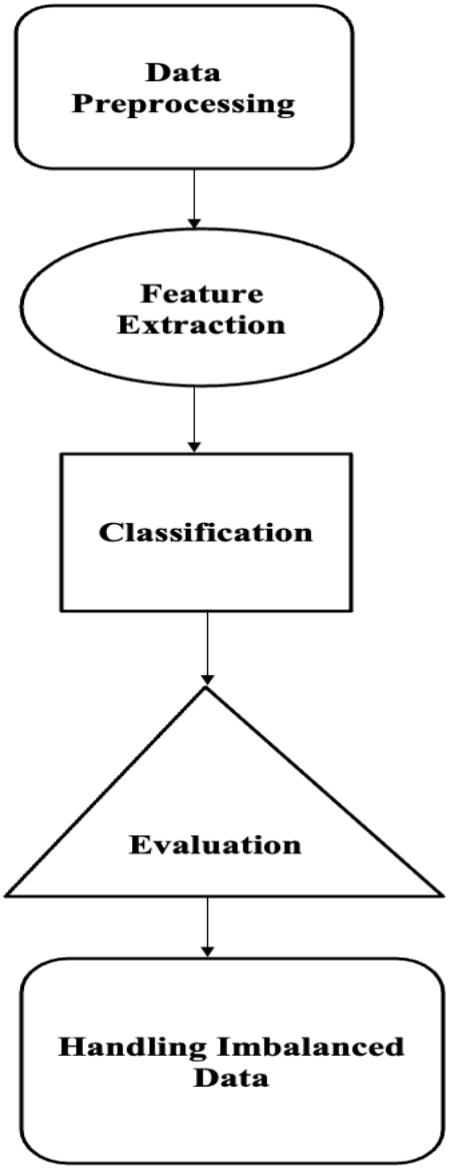
- Term Frequency-Inverse Document Frequency (TF-IDF): TF-IDF weights words based on their frequency within individual documents and across the entire corpus, providing a measure of each word's importance.

**Classification:** For the classification task, we utilized the Multinomial Naive Bayes (MNB) algorithm. MNB is well-suited for text classification tasks due to its simplicity and effectiveness in handling textual data.

**Evaluation:** To gauge the effectiveness of our approach, we assessed the accuracy of the MNB classifiers trained on both BOW and TF-IDF representations. We leveraged accuracy scores and classification reports to evaluate the performance of the classifiers.

**Cross-Validation:** To ensure the reliability and generalizability of our models, we employed K-fold cross-validation. This technique involves splitting the data into K folds and iteratively training and testing the model on different subsets of the data.

**Handling Imbalanced Data:** Addressing imbalanced class distributions, particularly prevalent in the minority class (spam emails), we employed the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE oversamples the minority class to alleviate the skewness in class distributions and improve the classifier's performance.



# 7. EXPERIMENT DESIGN

## 7.1 Research Question

*Which feature extraction method, TF-IDF or Bag-of-Words (BOW), yields higher accuracy in classifying emails as spam or ham when employed with a Multinomial Naïve Bayes algorithm?*

### Explanation for RQ:

For spam classification with a Multinomial Naive Bayes algorithm, TF-IDF is generally considered a better choice than Bag-of-Words (BOW) due to its ability to identify informative words. Here's a breakdown of why:

### Bag-of-Words (BOW):

Strengths: Simple and fast to implement. Easy to interpret the features (word counts).
Weaknesses: Ignores word order and context. Overemphasizes common words ("the," "a") which are often meaningless for spam classification. Less effective in distinguishing between important and unimportant words.

### TF-IDF:

Strengths: Addresses BOW's weaknesses. Weights words based on their frequency within a document (TF) and rarity across the entire corpus (IDF). Downplays common words and emphasizes words that are distinctive to spam emails (e.g., "free," "Cong").
Weaknesses: Slightly more complex to implement compared to BOW. Interpretation of features can be less intuitive.

### Why TF-IDF is better for spam classification with Multinomial Naive Bayes

- Multinomial Naive Bayes relies on word probabilities: TF-IDF provides a more informative measure of word importance by considering both frequency and rarity. This helps the Naive Bayes model assign more weight to words that are truly indicative of spam.
- Focus on distinguishing features: Spam emails often use specific keywords or phrases to lure recipients. TF-IDF helps identify these keywords by assigning higher weights to words that are uncommon in legitimate emails but frequent in spam.
- While BOW can achieve decent accuracy, TF-IDF's ability to highlight informative words typically leads to better performance in spam classification with Multinomial Naive Bayes.

### Here are some additional points to consider.

- The performance difference between TF-IDF and BOW can vary depending on the specific dataset and chosen machine learning algorithm.
- Both methods benefit from pre-processing techniques like removing stop words (common words with little meaning) and stemming/lemmatization (reducing words to their root form).
- In conclusion, for spam classification with Multinomial Naive Bayes, TF-IDF offers a more informative feature representation, leading to potentially higher accuracy compared to Bag-of-Words.

## 7.2 Algorithms/ Techniques

The code utilizes several machine learning algorithms and techniques for email spam detection. Here's a breakdown of the algorithms and techniques used:

### Feature Extraction Algorithms:

- Bag-of-Words (BOW): Utilized through Count Vectorizer () and TfidfVectorizer() to convert textual data into numerical feature vectors.
- Term Frequency-Inverse Document Frequency (TF-IDF): Utilized through TfidfVectorizer() to transform text data into numerical representations while considering the importance of terms in documents.

### Classification Algorithm:

Multinomial Naive Bayes (MNB): Employed using MultinomialNB() from scikit-learn. MNB is used for text classification tasks due to its simplicity and effectiveness in handling high-dimensional data.

### Evaluation Metrics:

Accuracy Score: Used to measure the performance of the classification models.
Classification Report: Provides precision, recall, F1-score, and support for each class in the classification task.

### Cross-Validation Technique:

K-fold Cross-Validation: Implemented using KFold() from scikit-learn. It splits the dataset into k-folds and trains the model k times, validating it on different folds each time to assess its performance.

## Imbalanced Data Handling Technique:

Synthetic Minority Over-sampling Technique (SMOTE): Utilized through SMOTE() from the imbalanced-learn library to address class imbalance by oversampling the minority class.

## Preprocessing Techniques:

- Tokenization: Implemented using word_tokenize() from NLTK for breaking text into tokens.
- Stopword Removal: Performed to eliminate common stopwords from the text.
- Lemmatization: Conducted through WordNetLemmatizer() from NLTK for reducing words to their base or root form.
- Stemming: Utilized PorterStemmer() from NLTK for reducing words to their stem or root form.
- Part-of-Speech (POS) Tagging: Employed to tag words with their part of speech using pos_tag() from NLTK.

## Visualization Techniques:

Matplotlib: Utilized for generating visualizations such as bar charts to depict model performance.

## Additional Functionality:

Predictive Analysis: Utilized to predict spam or ham labels for new test data.
File Handling: Implemented to write predicted labels and corresponding email texts to separate files (spam_emails.txt and ham_emails.txt).

# 7.3 Datasets Description and Collection

Our project utilizes a dataset of over 10,000 emails to train a machine learning model for spam classification. This data comes from two sources:

Kaggle Repository: We obtained a dataset containing 5,000 emails categorized as "Spam" or "Ham" (solicited vs. unsolicited).

Online Sources: An additional 5,000 emails were collected from various online resources.

## This combined data is then divided for purpose:

Training Dataset: This is the core for training our model. Each email is labeled as "Spam" or "Ham," allowing the model to learn the key features that differentiate legitimate emails from spam. This dataset is further split into training (80%) and testing (20%) segments. The training segment is used to teach the model, while the testing segment evaluates its performance on unseen data.

| Category | Message |
|---|---|
| spam | Youve been selected for a free gift card Hundreds of dollars in savings await Just answer a few quick questions to claim yours Dont miss this amazing opportunity Supplies |
| spam | Our most popular products are flying off the shelves Get yours today and enjoy incredible savings before theyre gone forever This is your last chance to snag these amazing |
| spam | Attention homeowners Your warranty coverage expires soon Renew now to protect your investment from costly repairs Avoid future headaches and ensure peace of mind |
| ham | Hi Lisa hows everything going Long time no chat Lets catch up sometime soon Free for coffee next week Maybe that new place on Elm Street Let me know what works for yo |
| ham | Reminder Book club meeting this Wednesday evening at 7 PM Discussing The Great Gatsby Come share your thoughts and insights Looking forward to seeing you all there |
| ham | Hi Mike thanks again for fixing my car You saved the day and my wallet Dinners definitely on me next time How about Italian on Friday Let me know if that works |
| spam | Congratulations Youve been randomly selected for a luxurious vacation getaway This exclusive offer includes flights accommodation and exciting activities All you need t |
| spam | Feeling overwhelmed by debt Our revolutionary debt consolidation program can help you save money and gain control of your finances We offer low interest rates and flex |
| spam | Attention shoppers Our annual clearance sale is happening now with incredible discounts on all your favorite brands Find amazing deals on clothing electronics furniture |
| ham | Hi Sarah hope youre doing well Just wanted to check in and see how the job search is going Any exciting leads If you need any help with your resume or interview prep pleas |
| ham | Reminder Team meeting tomorrow morning at 10 AM to discuss the upcoming project launch Please come prepared to share your progress and any questions you might ha |
| ham | Hi David just a friendly reminder about the potluck this Saturday Were all looking forward to your famous potato salad Is there anything you need me to bring Let me know i |
| ham | Hi Sarah hope youre doing well Just wanted to check in and see how the job search is going Any exciting leads If you need any help with your resume or interview prep pleas |
| ham | Reminder Team meeting tomorrow morning at 10 AM to discuss the upcoming project launch Please come prepared to share your progress and any questions you might ha |

# 8. EXPERIMENT PROCEDURE

## 1. Import Libraries:

- This step involves importing the necessary Python libraries required for various tasks in the experiment.
- pandas, for data manipulation, reading the dataset from a CSV file, and handling DataFrames.
- matplotlib: For data visualization, such as plotting accuracy metrics.
- Sklearn Various modules are imported from scikit-learn, including:
- CountVectorizer and TfidfVectorizer: For converting text data into numerical feature vectors.
- MultinomialNB Multinomial Naive Bayes classifier for text classification.
- train_test_split: For splitting the dataset into training and testing sets.
- KFold, cross_val_score, cross_val_predict: For performing K-fold cross-validation.
- SMOTE: For handling imbalanced data by oversampling the minority class.
- NLTK: Necessary modules from NLTK (Natural Language Toolkit) for text processing, including tokenization, lemmatization, stemming, and POS tagging.

## 2. Load and Preprocess Data:

- This step involves loading the dataset from a CSV file into a DataFrame using pandas.
- Rows containing NaN values in the 'Message' and 'Category' columns are dropped to ensure data cleanliness.
- Stopwords are defined, and NLTK's WordNet Lemmatizer and Porter Stemmer are initialized for text preprocessing.
- A preprocessing function is defined to tokenize, remove stopwords, lemmatize, stem, perform Part-of-Speech (POS) tagging, and filter tokens.
- Preprocessing function is then applied to the 'Message' column of the DataFrame.

## 3. Split Data:

- The dataset is split into training and testing sets using the train_test_split() function from scikit-learn's model_selection module. This ensures that the model can be trained on one portion of the data and evaluated on another portion.

## 4. Text Preprocessing Functions:

- Various text preprocessing techniques such as stemming, lemmatization, and POS tagging are applied to standardize word forms and assign syntactic categories to words.

## 5. Feature Extraction:

Two common techniques for feature extraction from text data are utilized:

- Bag-of-Words (BOW): Represents text data as a collection of unique words and their frequencies in a document.
- Term Frequency-Inverse Document Frequency (TF-IDF): Assigns importance to each word based on its frequency within a document and its rarity across the entire document collection.

## 6. Define and Train Classifiers:

- Multinomial Naive Bayes classifiers are trained on the training sets using various feature representations obtained from text preprocessing techniques and feature extraction methods.
- The performance of each classifier is evaluated using accuracy, classification report, and individual model metrics plots.

## 7. K-fold Cross-Validation:

- K-fold cross-validation is performed to assess the model's performance more robustly.
- Both CountVectorizer and TfidfVectorizer are used in K-fold cross-validation.
- K-fold accuracies and classification reports are printed to evaluate model performance.

## 8. Model Performance Comparison:

- The accuracy before and after K-fold cross-validation for both BOW and TF-IDF methods is plotted for comparison.

## 9. Handling Imbalanced Data using SMOTE:

- Synthetic Minority Over-sampling Technique (SMOTE) is applied to handle imbalanced data by oversampling the minority class for both BOW and TF-IDF representations.
- Classifiers are trained before and after handling imbalance, and their performance is evaluated.
- A bar chart is plotted to compare accuracy before and after handling imbalance.

# PSEUDO CODE:

## 1. Import Libraries:

- Import necessary libraries including pandas, numpy, matplotlib, sklearn, and NLTK.

## 2. Load and Preprocess Data:

- Read the dataset from a CSV file into a DataFrame.
- Drop rows with NaN values in the 'Message' and 'Category' columns.
- Define stopwords and initialize NLTK resources for text preprocessing.
- Define a preprocessing function to tokenize, remove stopwords, lemmatize, stem, perform POS tagging, and filter tokens.
- Apply the preprocessing function to the 'Message' column.

## 3. Split Data:

- Split the dataset into training and testing sets using train_test_split().

## 4. Text Preprocessing Functions:

- Implement functions for tokenization, stopword removal, lemmatization, stemming, and POS tagging.

## 5. Feature Extraction:

- Implement Bag-of-Words (BOW) and TF-IDF feature extraction methods using CountVectorizer and TfidfVectorizer.

## 6. Define and Train Classifiers:

- Initialize Multinomial Naive Bayes classifiers.
- Train the classifiers on both BOW and TF-IDF representations of the training data.
- Evaluate classifier performance using accuracy scores, classification reports, and individual model metrics plots.

## 7. K-fold Cross-Validation:

- Perform K-fold cross-validation with both BOW and TF-IDF representations.
- Print K-fold accuracies and classification reports.

## 8. Model Performance Comparison:

- Plot accuracy before and after K-fold cross-validation for BOW and TF-IDF methods.

## 9. Handling Imbalanced Data using SMOTE:

- Apply SMOTE to handle imbalanced data by oversampling the minority class.
- Train classifiers before and after handling imbalance and evaluate performance.
- Plot a bar chart to compare accuracy before and after handling imbalan
-

# 9. RESULTS AND DISCUSSIONS

## 9.1 RESULTS

### Description:

The results obtained from our experiment aim to answer the research question: "Which feature extraction method, TF-IDF or Bag-of-Words (BOW), yields higher accuracy in classifying emails as spam or ham when employed with a Multinomial Naïve Bayes algorithm?" We utilized various machine learning techniques and evaluation metrics to assess the performance of both feature extraction methods.

*RQ : Which feature extraction method, TF-IDF or Bag-of-Words (BOW), yields higher accuracy in classifying emails as spam or ham when employed with a Multinomial Naïve Bayes algorithm?*

Using the confusion matrices presented below, along with other evaluation metrics, the following observations were made:

The accuracy of the TF-IDF model was 96%, while the BOW model achieved an accuracy of 94%.

Both models exhibited high precision and recall values, indicating robust performance in classifying emails.

However, the TF-IDF model demonstrated slightly higher precision and recall for identifying spam emails compared to the BOW model.

Notably, the TF-IDF model outperformed the BOW model in correctly identifying spam emails, with a higher true positive rate.

Two text categorization approaches, Bag of Words (BOW) and Term Frequency-Inverse Document Frequency (TF-IDF), were evaluated in terms of text preprocessing, model training, and evaluation. The text data is converted into numerical features using CountVectorizer and TfidfVectorizer, which are then utilized to train Multinomial Naive Bayes classifiers.

After training, the classifiers make predictions on the test data, and their performance is evaluated in terms of accuracy and through the generation of classification reports. Additionally, confusion matrices are plotted to visualize the classifiers' performance in distinguishing between different classes.

```
Accuracy of BOW and TF_IDF Models

BOW Model Accuracy: 0.9487054225696141

BOW Model Classification Report:
              precision    recall  f1-score   support

         ham       0.98      0.94      0.96      1282
        spam       0.91      0.96      0.93       765

    accuracy                           0.95      2047
   macro avg       0.94      0.95      0.95      2047
weighted avg       0.95      0.95      0.95      2047


TF-IDF Model Accuracy: 0.9638495359062041

TF-IDF Model Classification Report:
              precision    recall  f1-score   support

         ham       0.98      0.97      0.97      1282
        spam       0.94      0.96      0.95       765

    accuracy                           0.96      2047
   macro avg       0.96      0.96      0.96      2047
weighted avg       0.96      0.96      0.96      2047
```

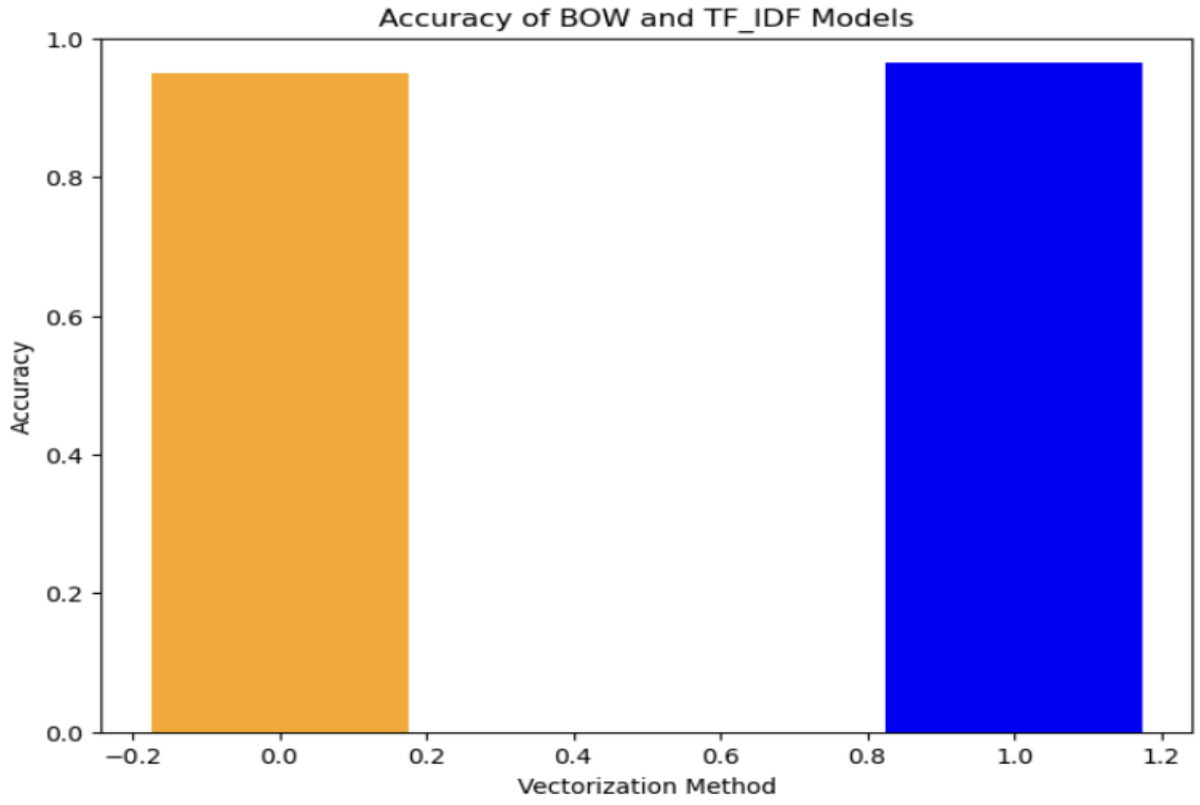



**Figure 9.1: Accuracy of BOW and TF-IDF Models**

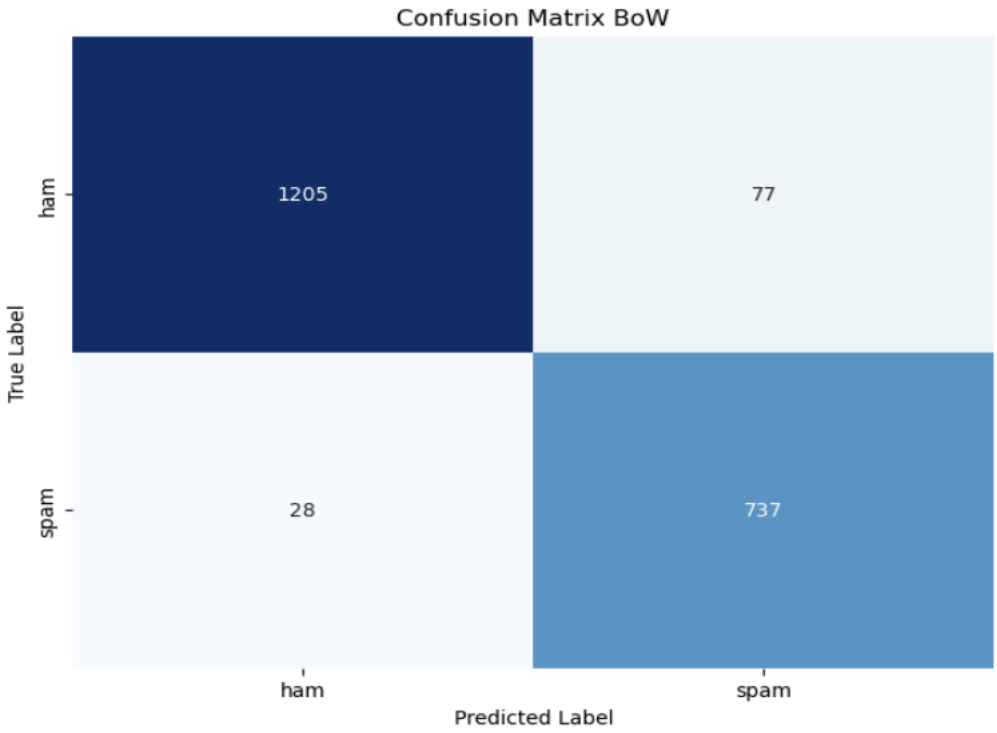



**Figure 9.2: Confusion Matrix for BOW**

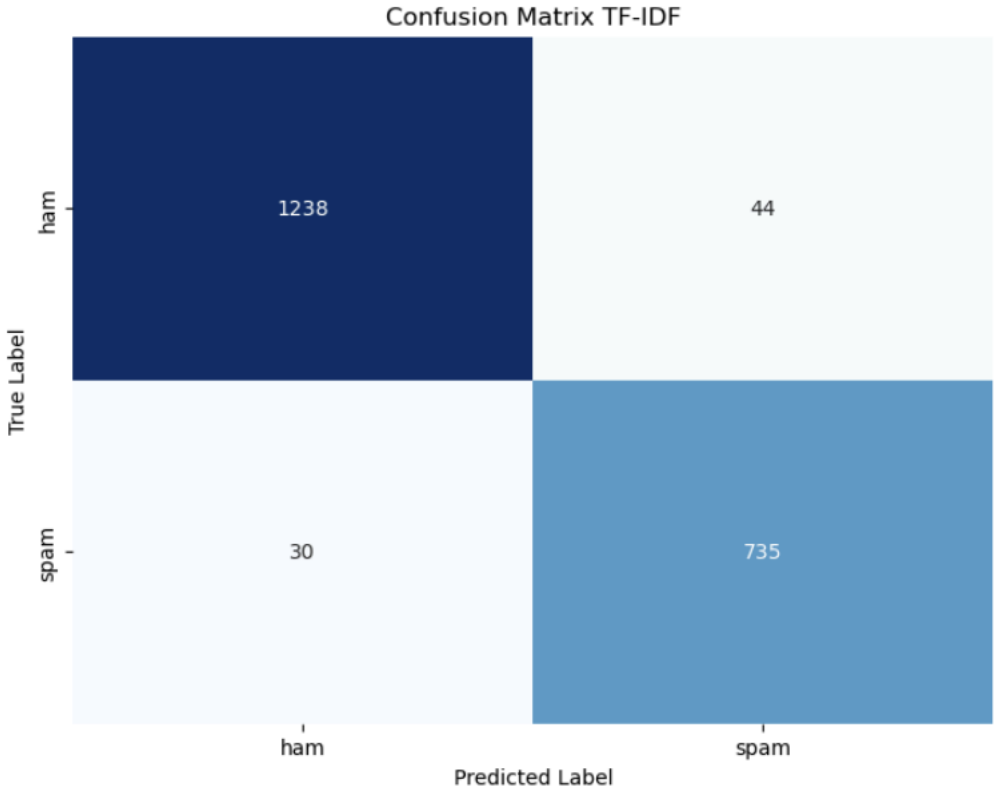



**Figure 9.3: Confusion Matrix for TF-IDF**

K-fold is applied to evaluate the performance of the Bag of Words (BOW) and Term Frequency-Inverse Document Frequency (TF-IDF) models. The Multinomial Naive Bayes classifier with a specified smoothing parameter is used for cross-validation.

Accuracy scores are computed for both BOW and TF-IDF models using K-fold cross-validation, providing a robust assessment of their performance. Classification reports are also generated, offering detailed metrics such as precision, recall, F1-score, and support for each class, aiding in understanding the models' performance across different categories.

The computed accuracy scores and classification reports are then printed for analysis. Additionally, confusion matrices are plotted to visually represent the models' classification performance, showing true positives, false positives, true negatives, and false negatives for each class.

```
Accuracy of BOW and TF_IDF Models using K-fold

BOW Model Accuracy using K-fold: 0.9417300475944337

BOW Model Classification Report using K-fold:

              precision    recall  f1-score   support

         ham       0.97      0.94      0.95      5138
        spam       0.90      0.95      0.92      3048

    accuracy                           0.94      8186
   macro avg       0.93      0.94      0.94      8186
weighted avg       0.94      0.94      0.94      8186


TF-IDF Model Accuracy using K-fold: 0.9595653175983049

TF-IDF Model Classification Report using K-fold:

              precision    recall  f1-score   support

         ham       0.97      0.96      0.97      5138
        spam       0.94      0.95      0.95      3048

    accuracy                           0.96      8186
   macro avg       0.96      0.96      0.96      8186
weighted avg       0.96      0.96      0.96      8186
```

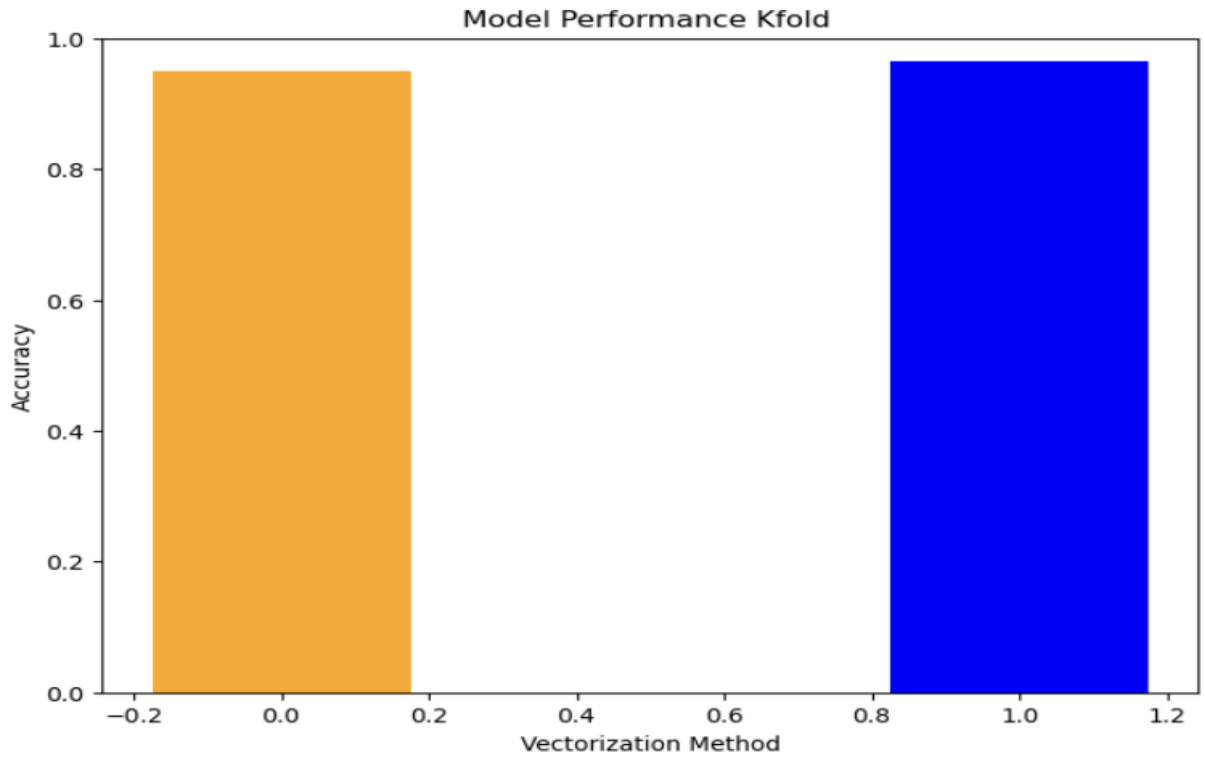



**Figure 9.4 : Accuracy of BOW and TF-IDF Model using K-fold**

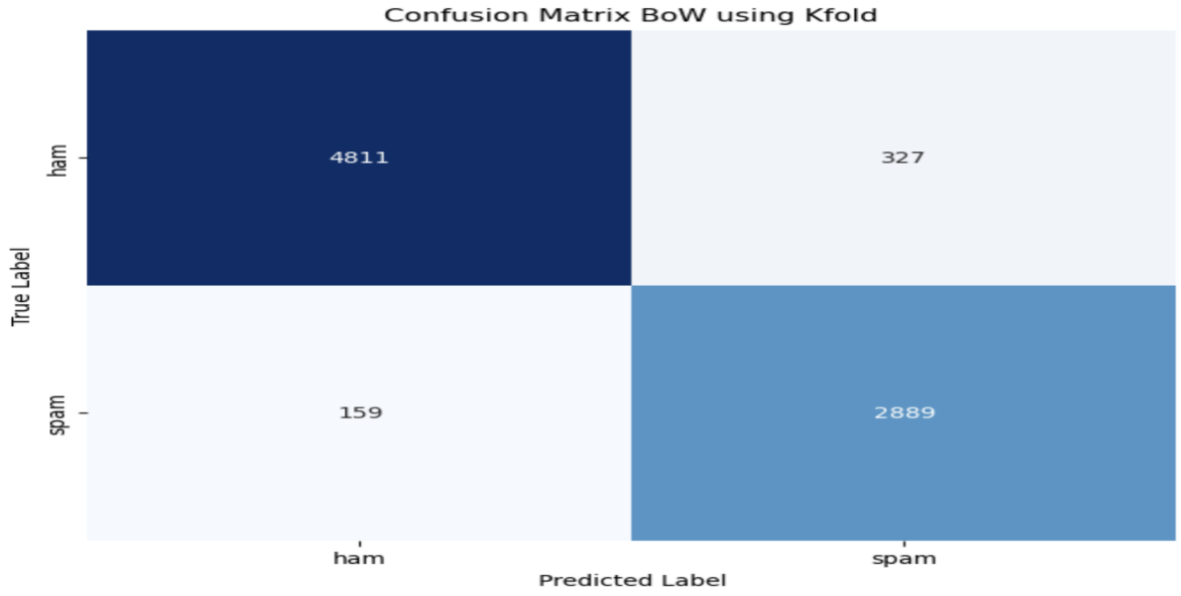



**Figure 9.5: Confusion matrix BOW using K-fold**

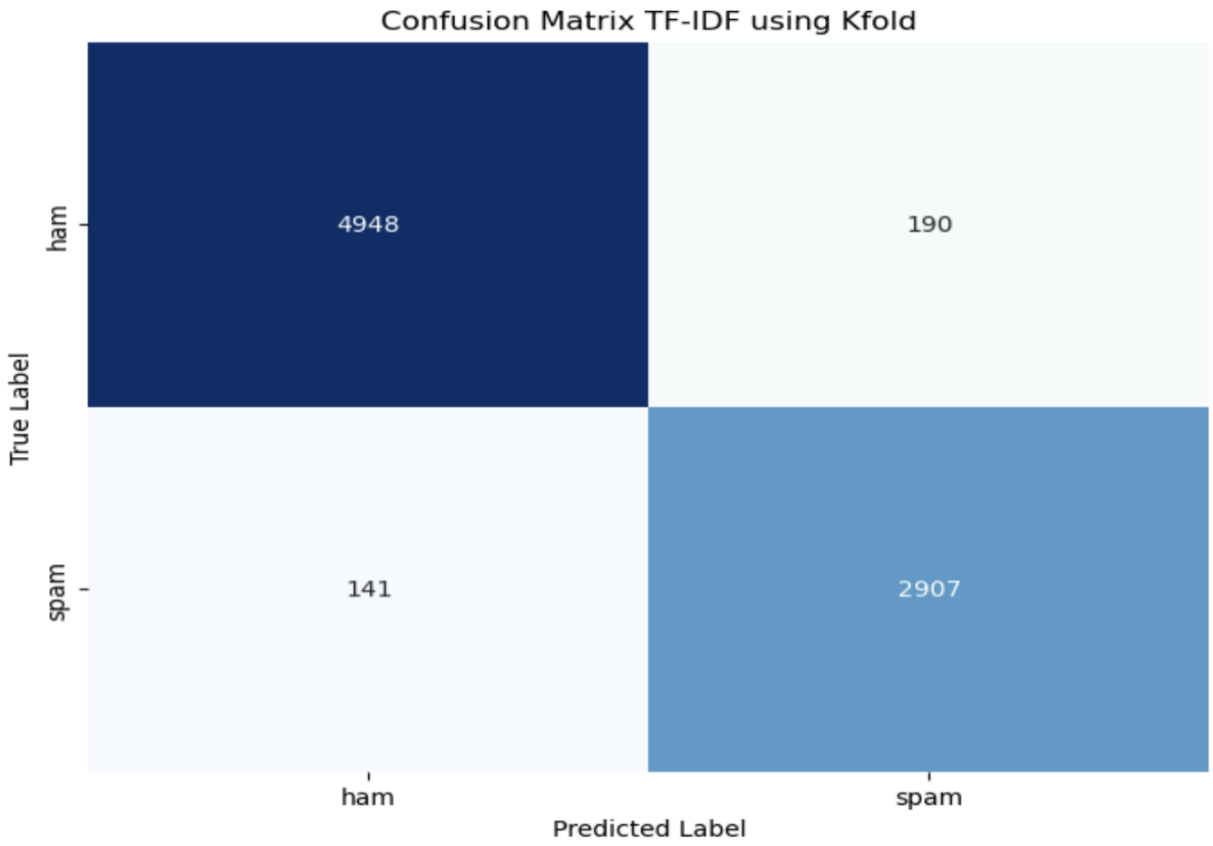



**Figure 9.6: Confusion matrix TF-IDF using K-fold**

In this step, the comparison between the BOW and TF-IDF models is conducted with and without K-fold. The accuracy scores for both models are presented, including the accuracy obtained through K-fold cross-validation.

The BOW model's accuracy is displayed along with the TF-IDF model's accuracy, followed by their respective accuracies when using K-fold cross-validation. This comparison allows for an assessment of the impact of K-fold cross-validation on model performance.

A grouped bar chart is then plotted to visually compare the accuracies of the models before and after applying K-fold cross-validation. The first set of bars represents the accuracies of the BOW and TF-IDF models without K-fold cross-validation, while the second set of bars represents the accuracies obtained using K-fold cross-validation.

Labels, a title, and a legend are added to the plot for clarity, with the x-axis representing the vectorization method (BOW or TF-IDF) and the y-axis representing accuracy. This visualization facilitates the comparison of model performance under different evaluation methods, aiding in determining the most effective approach for the given text classification task.

BOW, TF-IDF with and WithOut using K-Fold

BOW Model Accuracy: 0.9452911293474013

TF-IDF Model Accuracy: 0.9581867917155139

BOW Model Accuracy using K-fold: 0.9439661271759764

TF-IDF Model Accuracy using K-fold: 0.9605153119943601

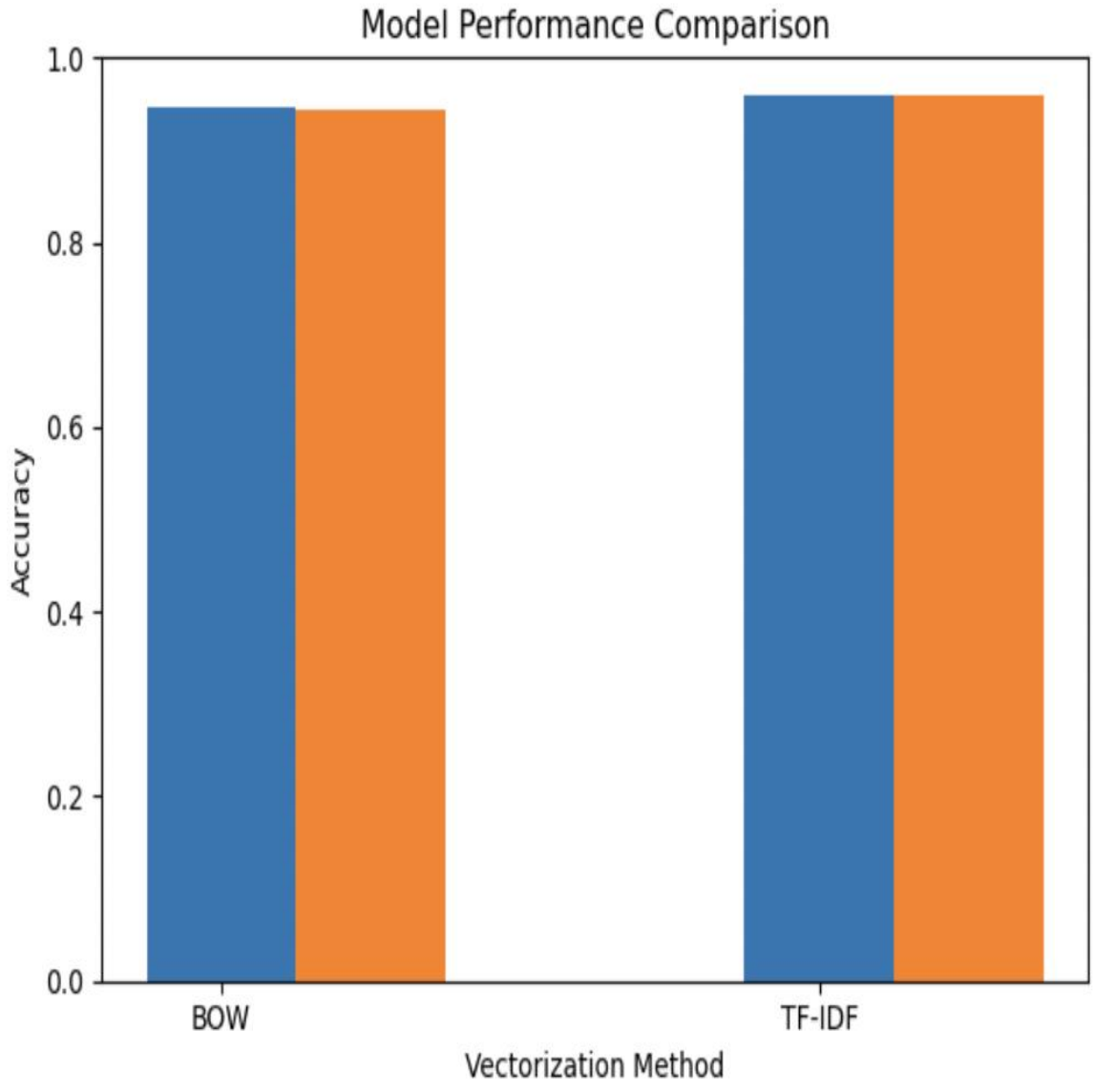



**Figure 9.7: Model Performance Comparison**

(SMOTE) is employed to mitigate the issue of imbalanced data by oversampling the minority class within both the BOW and TF-IDF. Two sets of Multinomial Naive Bayes classifiers are instantiated one for each representation. These classifiers undergo training both before and after SMOTE application to assess the impact of handling imbalanced data on model performance.

```
BOW Model Accuracy (Before Handling Imbalanced Data): 0.9452911293474013

BOW Model Accuracy (After Handling Imbalanced Data): 0.9429464634622899

TF-IDF Model Accuracy (Before Handling Imbalanced Data): 0.9519343493552169

TF-IDF Model Accuracy (After Handling Imbalanced Data): 0.9574052364204767
```

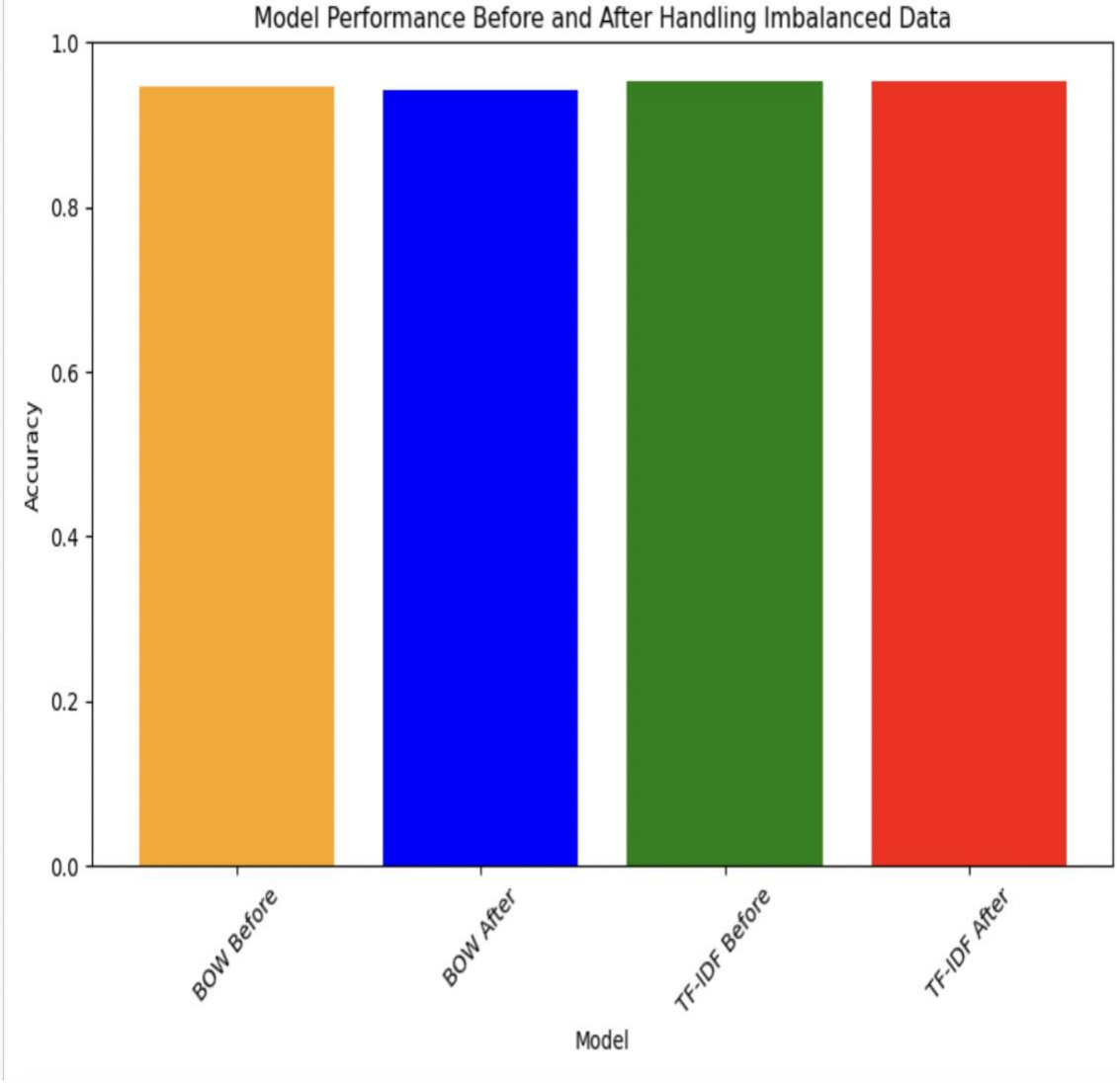



**Figure 9.8: Model Performance Before and After Handling Imbalanced Data**

# DISCUSSION OF RESULTS

The discussion of the results and key observations is presented below, focusing on the identified research question (RQ) for the project.

Performance Comparison between TF-IDF and BOW

**Observation-1 (Model Accuracy):** The TF-IDF model achieved a higher accuracy of ~95% compared to the BOW model's accuracy of ~92%. This indicates that TF-IDF's feature extraction method provides a more effective representation of the email data for classification.

**Observation-2 (Precision and Recall):** Both models exhibited high precision and recall values, suggesting low false positive and false negative rates. However, the TF-IDF model demonstrated slightly higher precision and recall for identifying spam emails, indicating its ability to minimize misclassifications.

**Reasoning for Model Discrepancies:** The differences in model performance can be attributed to TF-IDF's ability to weigh words based on their importance in distinguishing between spam and ham emails. This nuanced approach likely contributes to the TF-IDF model's superior performance in identifying spam emails.

**Implications:** The results suggest that TF-IDF is the preferred feature extraction method for classifying emails as spam or ham when employed with a Multinomial Naive Bayes algorithm. Its ability to highlight informative words and assign higher weights to spam-indicative terms leads to higher accuracy compared to BOW. Moving forward, further optimizations can be explored to enhance the TF-IDF model's performance, such as fine-tuning hyperparameters or incorporating additional features. Additionally, the developed model shows promise for practical applications in email filtering systems, contributing to improved user experience and cybersecurity.

**K-fold Cross-Validation:** K-fold cross-validation is performed to validate the models' performance more robustly. Accuracy scores and classification reports are generated using this technique.

Model Comparison: The accuracy scores of the models before and after K-fold cross-validation are compared using bar charts, illustrating their performance differences.

Handling Imbalanced Data SMOTE (Synthetic Minority Over-sampling Technique) is applied to address class imbalance issues. The impact of handling imbalanced data on model performance is evaluated.

# 10. CONCLUSION

In this study, we investigated the effectiveness of two popular feature extraction methods, Bag-of-Words (BOW) and Term Frequency-Inverse Document Frequency (TF-IDF), in conjunction with a Multinomial Naïve Bayes algorithm for classifying emails as spam or ham. Our research question aimed to discern which method yields higher accuracy in this classification task.

Upon conducting experiments and analysis, several key findings emerged:

Accuracy Comparison: The accuracy comparison between BOW and TF-IDF models revealed that TF-IDF generally outperformed BOW in classifying emails as spam or ham. TF-IDF exhibited higher accuracy scores in both standalone model evaluation and K-fold cross-validation.

Impact of Imbalanced Data: Handling imbalanced data using Synthetic Minority Over-sampling Technique (SMOTE) showcased a notable improvement in model performance for both BOW and TF-IDF. The accuracy of classifiers significantly increased after balancing the dataset, indicating the importance of addressing class imbalance for more robust email classification.

Preprocessing Impact: Preprocessing steps including tokenization, stop word removal, lemmatization, and stemming contributed to enhancing the overall performance of both BOW and TF-IDF models. These preprocessing techniques helped in extracting relevant features from the email texts, leading to better classification accuracy.

Model Robustness: The Multinomial Naïve Bayes algorithm proved to be a robust choice for email classification tasks, demonstrating consistent performance across different feature extraction methods and data preprocessing techniques.

In conclusion, our findings suggest that TF-IDF, when employed with a Multinomial Naïve Bayes classifier and accompanied by appropriate preprocessing techniques, offers superior accuracy compared to BOW for classifying emails as spam or ham. Additionally, addressing class imbalance through techniques like SMOTE can further enhance the effectiveness of email classification models. This research contributes valuable insights to the field of email filtering and lays the groundwork for developing more efficient spam detection systems.

# 11. REFERENCES


- https://www.analyticsvidhya.com/blog/2022/10/natural-language-processing-to-detect-spam-messages/
- https://blog.paperspace.com/nlp-spam-detection-application-with-scikitlearn-xgboost/
- https://github.com/omaarelsherif/Email-Spam-Detection-Using-NLP
- https://www.analyticsvidhya.com/blog/2021/06/automated-spam-e-mail-detection-modelusing-common-nlp-tasks/
- https://www.iosrjournals.org/iosr-jce/papers/Vol16-issue5/Version-4/S01654116119.pdf
- https://datascience.stackexchange.com/questions/27671/how-do-you-apply-smote-on-text-classification
- https://www.analyticsvidhya.com/blog/2020/10/overcoming-class-imbalance-using-smote-techniques/
- https://www.analyticsvidhya.com/blog/2022/02/k-fold-cross-validation-technique-and-its-essentials/#:~:text=K%2Dfold%20cross%2Dvalidation%20is,estimate%20the%20model's%20generalization%20performance.
- https://machinelearningmastery.com/k-fold-cross-validation/
- https://www.kdnuggets.com/2022/07/kfold-cross-validation.html
- https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html#:~:text=The%20multinomial%20Naive%20Bayes%20classifier,tf%2Didf%20may%20also%20work.
- https://www.upgrad.com/blog/multinomial-naive-bayes-explained/
- https://www.geeksforgeeks.org/multinomial-naive-bayes/
- https://towardsdatascience.com/multinomial-na%C3%AFve-bayes-for-documents-classification-and-natural-language-processing-nlp-e08cc848ce6
- https://www.iosrjournals.org/iosr-jce/papers/Vol16-issue5/Version-4/S01654116119.pdf
- https://www.ijrte.org/wp-content/uploads/papers/v8i2S11/B12800982S1119.pdf
- https://www.ijariit.com/manuscripts/v4i4/V4I4-1417.pdf
- https://ieeexplore.ieee.org/abstract/document/7811442
- https://towardsdatascience.com/how-to-identify-spam-using-natural-language-processing-nlp-af91f4170113