**Email Spam Detection**

****

**A Project Report**

Submitted for Minor Project – CS6490 of 6th Semester for the partial fulfilment of the requirement for the award of the degree

of

**Bachelor of Technology**

in

**Computer Science and Engineering**

submitted by

**Piyush Kumar (2006008)**

**Pradeep Kumar Singh (2006010)**

**Akshat Singhal (2006059)**

Under the supervision of

**Prof. Maheshwari Prasad Singh**

HOD

CSE Department

NIT Patna

**Department of Computer Science and Engineering**

**National Institute of Technology, Patna – 800005**

Jan- June 2023

# CERTIFICATE

This is to certify that Piyush Kumar with Roll No. 2006008, Pradeep Kumar Singh with Roll No. 2006010, Akshat Singhal with Roll No. 2006059 has carried out the Minor Project (CS6490) entitled "Email Spam Detection" during 6th semester under the guidance of Prof.Maheshwari Prasad Singh, HOD, CSE Department, in partial fulfilment of the requirements for the award of Bachelor of Technology degree in the Department of Computer Science & Engineering, National Institute of Technology, Patna.

………………………………..

**Prof. Maheshwari Prasad Singh**

HOD, CSE Department

NIT Patna

# DECLARATION

We, the students of 6th semester, hereby declare that this project entitled "**Email Spam Detection**" has been carried out by us in the Department of Computer Science and Engineering of National Institute of Technology Patna under the guidance of Prof. **Maheshwari Prasad Singh,** HOD of Computer Science and Engineering, NIT Patna. No part of this project has been submitted for the award of degree to any other Institute.

|  |  |
| --- | --- |
| **Name** | **Roll No.** |
| Piyush Kumar | 2006008 |
| Pradeep Kumar Singh | 2006010 |
| Akshat Singhal | 2006059 |

|  |  |
| --- | --- |
| **Place** | **Date** |
| NIT Patna | 4th May 2023 |

# ACKNOWLEDGEMENT

We take this opportunity to express our profound gratitude and deep regards to Prof. Prof. Maheshwari Prasad Singh, for his exemplary guidance, monitoring and constant encouragement throughout the course of this project. We also extend our heartfelt thanks to the entire CSE department of NIT Patna, for providing the necessary technical facilities and the technical environment which supported us to perform to the very best of our potential. The motivation that we gained for this project, shall go a long way in learning and leveraging these technologies further, and incorporating them in our project work in the future. We are extremely humbled and gratified for the opportunity and exposure provided to us in Machine learning and generative adversarial networks.

1. Piyush Kumar
2. Pradeep Kumar Singh
3. Akshat Singhal

Contents

[CERTIFICATE 2](#_Toc135176598)

[DECLARATION 3](#_Toc135176599)

[ACKNOWLEDGEMENT 4](#_Toc135176600)

[ABSTRACT 7](#_Toc135176601)

[1. Introduction 8](#_Toc135176602)

[1.1 Emails: 8](#_Toc135176603)

[1.2 Email Spam Detection: 8](#_Toc135176604)

[2. Motivation 9](#_Toc135176605)

[2.1 Proliferation of Email Spam: 9](#_Toc135176606)

[2.2 Impact on User Experience: 9](#_Toc135176607)

[2.3 Security Concerns: 9](#_Toc135176608)

[3. Proposed Framework: 10](#_Toc135176609)

[3.1 Classifier Selection: 10](#_Toc135176610)

[3.2 Feature Extraction: 10](#_Toc135176611)

[3.3 Training and Testing: 10](#_Toc135176612)

[3.4 Comparative Analysis: 11](#_Toc135176613)

[4. Workflow: 12](#_Toc135176614)

[4.1 Spam Email Data Collection: 12](#_Toc135176615)

[4.2 Data Cleaning: 13](#_Toc135176616)

[4.2.1 Handling Missing Values: 13](#_Toc135176617)

[4.2.2 Label Encoding: 14](#_Toc135176618)

[4.2.3 Removing Duplicates: 15](#_Toc135176619)

[4.3 Exploratory Data Analysis (EDA): 15](#_Toc135176620)

[4.3.1 Number of Characters: 16](#_Toc135176621)

[4.3.2 Number of Words: 16](#_Toc135176622)

[4.3.3 Number of Sentences: 16](#_Toc135176623)

[5. Text Pre-processing: 18](#_Toc135176624)

[5.1 Lowercase Conversion: 18](#_Toc135176625)

[5.2 Tokenization: 18](#_Toc135176626)

[5.3 Removal of Special Characters, Stop Words, and Punctuation: 18](#_Toc135176627)

[5.4 Stemming: 19](#_Toc135176628)

[6. Model Building: 20](#_Toc135176629)

[6.1 Feature Extraction: 20](#_Toc135176630)

[6.1.1 Count Vectorizer: 20](#_Toc135176631)

[6.1.2 TF-IDF (Term Frequency-Inverse Document Frequency): 21](#_Toc135176632)

[6.1.3 TF-IDF with Limited Features: 21](#_Toc135176633)

[6.2 Model Selection and Implementation: 22](#_Toc135176634)

[6.2.1 Naive Bayes Classifiers: 22](#_Toc135176635)

[6.2.2 Support Vector Classifier (SVC): 25](#_Toc135176636)

[6.2.3 Decision Trees: 26](#_Toc135176637)

[7. Comparative Analysis and Results 28](#_Toc135176638)

[7.1 Accuracy: 28](#_Toc135176639)

[7.2 Precision: 28](#_Toc135176640)

[7.3 Recall (Sensitivity or True Positive Rate): 28](#_Toc135176641)

[7.4 The significance of these parameters in evaluating the models: 28](#_Toc135176642)

[8. CONCLUSION 34](#_Toc135176643)

[REFERENCES 35](#_Toc135176644)

# ABSTRACT

The increasing volume of spam emails poses a significant challenge to effective communication and information management. To address this issue, the present project focuses on the development of an email spam detection model using machine learning techniques. The project leverages a publicly available dataset of spam emails and employs a comprehensive workflow that includes data collection, data cleaning, exploratory data analysis, feature extraction, text pre-processing, and model building. Three types of Naive Bayes classifiers, Support Vector Classifier (SVC), and Decision Tree Classifier are utilized for model building. The models are evaluated based on key performance metrics such as accuracy, precision, and recall. The comparative analysis reveals insights into the strengths and weaknesses of each model, enabling the identification of the most suitable algorithm for email spam classification. Furthermore, the requirements of minimizing false positives and effectively detecting spam emails are taken into consideration during the evaluation process. The findings from this project can aid in the development of robust email spam detection systems, enhancing communication security and efficiency.

Project Repository: The source code and related materials for this project can be accessed on GitHub at the following link:

<https://github.com/piyusheverhard/email-spam-detection>

# 1. Introduction

The introduction section provides an overview of emails and the significance of email spam detection. It sets the context for the project and outlines the goals and objectives.

## 1.1 Emails:

Emails have become an indispensable form of communication in the digital age. They enable individuals and organizations to exchange messages, documents, and other information efficiently and globally. However, the widespread use of emails has also led to an increase in unwanted and unsolicited messages, commonly known as email spam.

## 1.2 Email Spam Detection:

Email spam detection refers to the process of identifying and filtering out unwanted and potentially harmful emails from a user's inbox. The primary objective of spam detection is to minimize the impact of spam on users by ensuring that genuine and important messages reach their intended recipients while filtering out irrelevant or malicious content.

# 2. Motivation

The motivation section delves into the reasons that prompted the undertaking of the email spam detection project. It highlights the problems and challenges associated with email spam and the significance of developing effective detection mechanisms.

## 2.1 Proliferation of Email Spam:

The rapid growth of the internet and the increasing number of email users have resulted in a corresponding surge in email spam. These unsolicited messages often contain fraudulent schemes, phishing attempts, malware, or irrelevant promotional content. Email spam not only wastes users' time and resources but also poses significant security risks.

## 2.2 Impact on User Experience:

Email spam undermines the efficiency and usability of email systems. Users are overwhelmed by the sheer volume of spam messages, which can lead to missed important emails, reduced productivity, and frustration. There is a growing need for robust and accurate email spam detection techniques to improve the overall user experience and ensure the delivery of legitimate emails.

## 2.3 Security Concerns:

Email spam serves as a medium for various cyber threats, including phishing attacks, identity theft, and the distribution of malware. It is crucial to identify and prevent such malicious activities to safeguard users' personal information, financial assets, and sensitive data. The project aims to address these security concerns by implementing effective email spam detection algorithms.

# 3. Proposed Framework:

The proposed framework encompasses the development and evaluation of an email spam detection system using a combination of three classifiers: Naive Bayes, Support Vector Machines (SVM), and Decision Trees. This approach aims to leverage the strengths of each classifier to improve the accuracy and robustness of the spam detection system.

## 3.1 Classifier Selection:

The choice of classifiers is based on their proven effectiveness in text classification tasks and their suitability for email spam detection. Naive Bayes classifiers are well-known for their simplicity, speed, and ability to handle high-dimensional data. SVMs are powerful in handling complex decision boundaries and have shown promising results in various classification problems. Decision trees provide interpretability and are capable of capturing complex feature interactions.

## 3.2 Feature Extraction:

Before applying the classifiers, an essential step involves feature extraction from the email data. This typically involves transforming the raw email text into a numerical representation that can be understood by the classifiers. Common feature extraction techniques include bag-of-words, TF-IDF (Term Frequency-Inverse Document Frequency), and word embeddings.

## 3.3 Training and Testing:

The proposed framework involves training each classifier using a labelled dataset consisting of spam and legitimate emails. The dataset is divided into training and testing subsets, ensuring that the classifiers are evaluated on unseen data. During the training phase, the classifiers learn the underlying patterns and characteristics of spam and legitimate emails.

## 3.4 Comparative Analysis:

After training the classifiers, a comparative analysis is performed to evaluate their performance. Several evaluation metrics, such as accuracy, precision, recall, and F1 score, are calculated for each classifier. This analysis provides insights into the strengths and weaknesses of each classifier and helps identify the most effective approach for email spam detection.

# 4. Workflow:

The workflow section outlines the sequential steps involved in the email spam detection project. It provides a comprehensive overview of the entire process, from data collection to model evaluation. Each step is crucial in ensuring the accuracy and effectiveness of the spam detection system.

The workflow section encompasses the following stages:

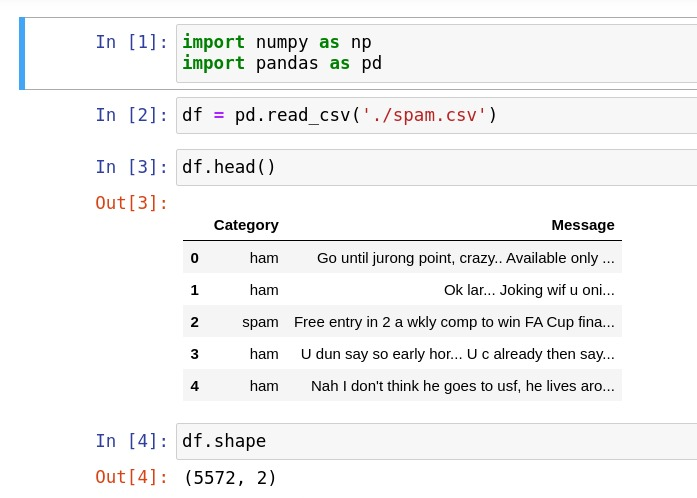
## 4.1 Spam Email Data Collection:

The selection and collection of an appropriate dataset are crucial for training and evaluating an effective email spam detection model. In this project, a publicly available dataset called the "spam email dataset" from Kaggle was utilized.

The dataset consists of 5572 rows and 2 columns. The first column contains labels indicating whether an email is classified as "ham" (legitimate) or "spam", while the second column contains the actual email content. This dataset provides a diverse range of emails, including both legitimate and spam examples.

The importance of a well-curated dataset cannot be overstated. It serves as the foundation for training the email spam detection model and plays a vital role in determining its accuracy and performance. A high-quality dataset with a balanced distribution of spam and legitimate emails allows the model to learn and distinguish the characteristics and patterns associated with each class.

By using a publicly available dataset, the project benefits from the collective efforts of data contributors and the wider community. This dataset provides a starting point for developing and evaluating the email spam detection model, allowing for comparison and benchmarking against other models and techniques. It is worth noting that the dataset selection process should consider factors such as data size, representativeness, label quality, and any specific domain considerations. Additionally, data pre-processing steps, such as removing duplicates and handling missing values, might be required to ensure the dataset's cleanliness and reliability.

****

## 4.2 Data Cleaning:

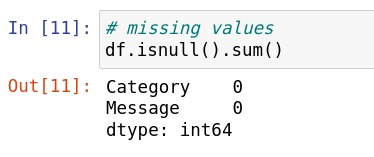
Data cleaning is a crucial step in preparing the dataset for further analysis and model training. It involves identifying and handling inconsistencies, errors, missing values, and duplicates in the dataset. The importance of data cleaning lies in ensuring the quality, reliability, and consistency of the data, which directly impact the accuracy and performance of the email spam detection model.

**Methodology:**

The data cleaning process typically follows a systematic methodology that includes the following steps:

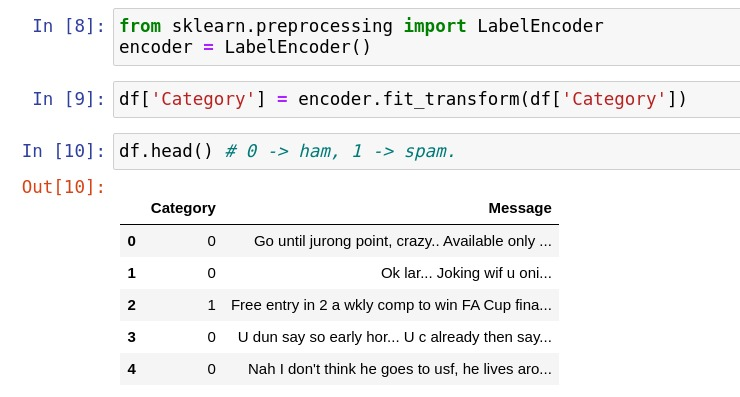
### 4.2.1 Handling Missing Values:

Missing values in the dataset can occur due to various reasons, such as incomplete data or data extraction errors. These missing values need to be addressed before proceeding with the analysis. Depending on the extent and nature of missing values, techniques like imputation (replacing missing values with estimated values) or deletion (removing rows or columns with missing values) can be applied.



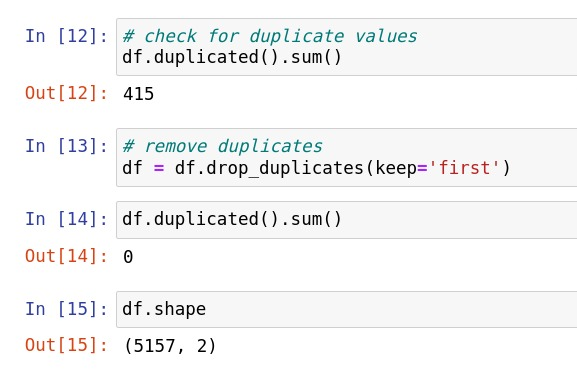
### 4.2.2 Label Encoding:

Label encoding is performed to transform the "ham" and "spam" labels into numerical values. By assigning 0 to "ham" and 1 to "spam", the model can effectively learn and differentiate between legitimate and spam emails during training. This label encoding process ensures a consistent and machine-readable representation of the classes, facilitating accurate predictions and evaluations in the subsequent stages of the project.



### 4.2.3 Removing Duplicates:

Duplicates in the dataset can skew the analysis and model training process. It is crucial to identify and remove any duplicate records to ensure each email is represented only once in the dataset. Duplicate removal can be performed based on various criteria, such as comparing email content, email headers, or a combination of attributes.



## 4.3 Exploratory Data Analysis (EDA):

Exploratory Data Analysis is a crucial step in understanding the characteristics and patterns present in the dataset. In the context of spam email detection, EDA provides valuable insights into the structure and composition of spam emails. In this project, the dataset was analysed based on the number of characters, words, and sentences in the spam emails.

Analysing the dataset on the basis of the following metrics provides a deeper understanding of the spam email content:

### 4.3.1 Number of Characters:

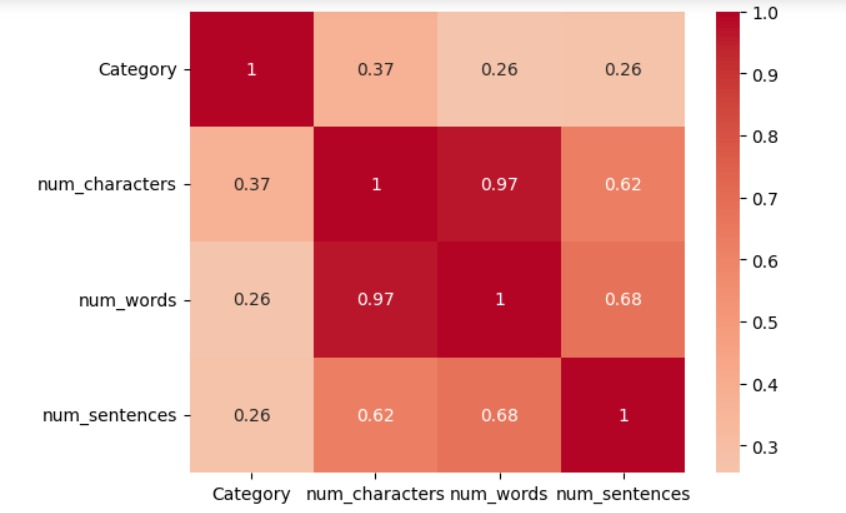
By examining the number of characters in spam emails, insights can be gained regarding the length and complexity of the text. Longer spam emails may indicate attempts to deceive recipients with detailed information or persuasive tactics. Conversely, shorter emails might employ concise, attention-grabbing techniques. Analysing the distribution of character counts helps in identifying any significant trends or outliers.

### 4.3.2 Number of Words:

The number of words in spam emails offers insights into the complexity and linguistic characteristics of the text. Analysing the distribution of word counts helps in understanding the average length of spam emails. Unusually high word counts might indicate content designed to confuse spam filters or deceive recipients, while abnormally low word counts may suggest the use of brief, targeted messages. EDA on word counts helps uncover important patterns and variations within the spam email dataset.

### 4.3.3 Number of Sentences:

Analysing the number of sentences in spam emails provides insights into the structure and organization of the text. Understanding the distribution of sentence counts helps identify any patterns related to spam email composition. Emails with a high number of sentences may employ multiple tactics, such as social engineering or extensive product descriptions, to trick recipients. Conversely, emails with fewer sentences might rely on a more concise and straightforward approach. EDA on sentence counts helps reveal key characteristics of spam email writing style.



During the exploratory data analysis, a Pearson correlation analysis was conducted to examine the relationship between the properties (number of words, number of characters, and number of sentences) and spam emails. The correlation coefficients revealed a moderate positive linear relationship between the number of words (correlation coefficient of 0.37), the number of characters (correlation coefficient of 0.26), and the number of sentences (correlation coefficient of 0.26) with the presence of spam emails. However, these correlation values were not considered high enough to be deemed significant predictors for spam email detection in this project.

# 5. Text Pre-processing:

Text pre-processing is a critical step in preparing the email text data for analysis and model training. It involves transforming the raw text into a clean and standardized format, removing noise, and reducing the dimensionality of the data. In this project, several pre-processing techniques were applied to the email text data, including lowercase conversion, tokenization, removal of special characters, stop words, punctuation, and stemming.

## 5.1 Lowercase Conversion:

The first step in text pre-processing is converting all text to lowercase. This step ensures that the text is standardized and eliminates any inconsistencies due to varying capitalization. Converting all text to lowercase allows for better matching of words and reduces the complexity of the vocabulary.

## 5.2 Tokenization:

Tokenization is the process of breaking down the text into individual words or tokens. This step segments the text into meaningful units, such as words or n-grams. Tokenization facilitates further analysis, as it enables the model to understand the structure and context of the text by treating each token as a separate entity.

## 5.3 Removal of Special Characters, Stop Words, and Punctuation:

Special characters, stop words, and punctuation marks do not typically carry significant meaning in text analysis and can introduce noise or unwanted variations. Hence, these elements are removed from the text. Special characters and punctuation marks are eliminated to ensure that only meaningful words and tokens are retained. Stop words, which are commonly occurring words like "a," "the," or "is," are removed as they do not contribute much to the overall meaning of the text.

## 5.4 Stemming:

Stemming is the process of reducing words to their root or base form by removing suffixes or prefixes. This step reduces the dimensionality of the vocabulary by combining words with similar meanings into a common root word. Stemming helps in capturing the essence of words and consolidating them, thus improving the model's ability to recognize patterns and generalize across variations of words.

We have created a function to perform text pre-processing on the email data.



# 6. Model Building:

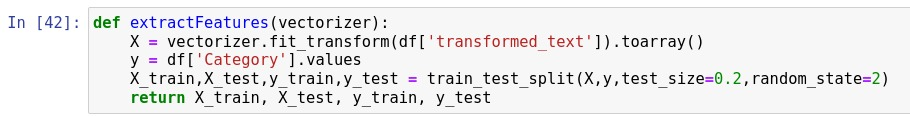
In the email spam detection project, model building involves the selection and implementation of various algorithms to train and evaluate the email spam detection model. Additionally, feature extraction techniques play a crucial role in transforming the textual data into a numerical representation that can be utilized by the machine learning algorithms.

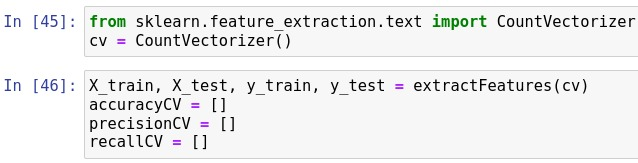
## 6.1 Feature Extraction:

Feature extraction is a critical step in converting the raw text data into a structured numerical format. In this project, three feature extraction methods were employed:

### 6.1.1 Count Vectorizer:

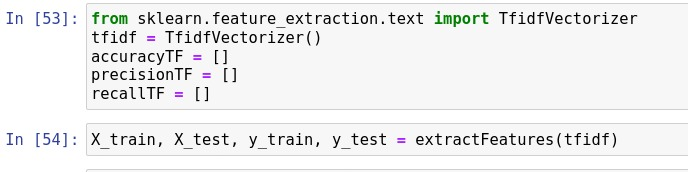
Count Vectorizer is a popular technique that converts a collection of text documents into a matrix of token counts. It represents each document as a vector of term frequencies. The Count Vectorizer extracts features by counting the occurrences of each word in the dataset, constructing a vocabulary, and generating a sparse matrix representation. This technique is effective in capturing the importance of individual words in the text.





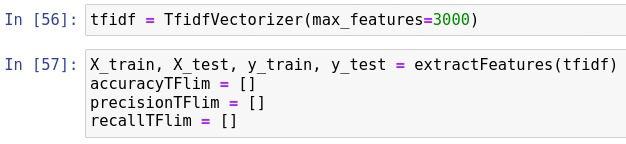
### 6.1.2 TF-IDF (Term Frequency-Inverse Document Frequency):

TF-IDF is a widely used feature extraction technique that evaluates the importance of a term within a document relative to its occurrence across all documents in the dataset. It combines the term frequency (TF) and inverse document frequency (IDF) to calculate a weight for each term in the corpus. The TF-IDF representation helps in emphasizing words that are highly relevant to a specific document but occur infrequently in the entire dataset.



### 6.1.3 TF-IDF with Limited Features:

To reduce the dimensionality of the feature space and enhance computational efficiency, the TF-IDF feature extraction technique was further modified by limiting the maximum number of features to 3000. This approach selects the top 3000 most significant terms based on their TF-IDF scores, discarding less informative terms. By restricting the feature space, it helps in improving model performance and reducing overfitting.



## 6.2 Model Selection and Implementation:

To build an effective email spam detection model, three types of Naive Bayes classifiers (MultinomialNB, BernoulliNB, and GaussianNB), Support Vector Classifier (SVC), and decision trees were employed. Each algorithm has its own mathematical equations and assumptions:

### 6.2.1 Naive Bayes Classifiers:

Naive Bayes classifiers are probabilistic models based on Bayes' theorem. They assume that the features are conditionally independent of each other given the class label. Although this assumption is often violated in real-world scenarios, Naive Bayes classifiers have shown to perform well in practice, especially for text classification tasks such as email spam detection.

Mathematical Equation:

The mathematical equation underlying the Naive Bayes classifiers is Bayes' theorem:

**P(Y|X) = (P(X|Y) \* P(Y)) / P(X)**

Where:

*P(Y|X) represents the probability of the class label Y given the features X.*

*P(X|Y) is the probability of observing the features X given the class label Y.*

*P(Y) denotes the prior probability of the class label Y.*

*P(X) is the probability of observing the features X.*

#### 6.2.1.1 MultinomialNB:

MultinomialNB is a Naive Bayes classifier that assumes that the features (in this case, word frequencies) follow a multinomial distribution. It is commonly used for text classification tasks where features represent word counts or frequencies.

Mathematical Equation:

For MultinomialNB, the probability of a class label Y given the features X can be calculated using:

**P(Y|X) ∝ P(Y) \* ∏(P(x\_i|Y))**

Where:

*P(x\_i|Y) is the conditional probability of observing feature x\_i given the class label Y.*

In the context of email spam detection, MultinomialNB can estimate the probabilities of an email being spam or legitimate based on the frequencies of words present in the email. It learns the probabilities from the training data and assigns the class label with the highest probability to new, unseen emails.

#### 6.2.1.2 BernoulliNB:

BernoulliNB is a Naive Bayes classifier that assumes binary features (presence or absence of a feature). It is suitable for text classification tasks where features are represented as binary values, such as the presence or absence of specific words.

Mathematical Equation:

For BernoulliNB, the probability of a class label Y given the features X can be calculated using:

**P(Y|X) ∝ P(Y) \* ∏(P(x\_i|Y))**

Where:

*P(x\_i|Y) is the conditional probability of observing feature x\_i given the class label Y.*

In the context of email spam detection, BernoulliNB can estimate the probabilities of an email being spam or legitimate based on the presence or absence of specific words. It learns the probabilities from the training data and assigns the class label with the highest probability to new, unseen emails.

#### 6.2.1.3 GaussianNB:

GaussianNB is a Naive Bayes classifier that assumes that the features follow a Gaussian (normal) distribution. It is suitable for continuous or real-valued features. In the context of email spam detection, GaussianNB is not commonly used directly on the text data itself, but it can be utilized when additional numerical features (e.g., length of email, number of URLs) are considered along with the text features.

Mathematical Equation:

For GaussianNB, the probability of a class label Y given the features X can be calculated using:

**P(Y|X) ∝ P(Y) \* ∏(P(x\_i|Y))**

Where:

*P(x\_i|Y) is the probability density function of observing feature x\_i given the class label Y, assuming a Gaussian distribution.*

In the context of email spam detection, GaussianNB can estimate the probabilities of an email being spam or legitimate based on additional numerical features. It learns the probabilities from the training data and assigns the class label with the highest probability to new, unseen emails.

### 6.2.2 Support Vector Classifier (SVC):

Support Vector Classifier (SVC) is a powerful machine learning algorithm used for classification tasks. It aims to find an optimal hyperplane that maximally separates the data points belonging to different classes. In the case of email spam classification, SVC can be applied to separate spam emails from legitimate emails based on the given features.

Mathematical Equations:

The objective of SVC is to find a hyperplane that maximally separates the data points. The equation of a hyperplane in a binary classification scenario is defined as:

**w · x + b = 0**

*where:*

*w is the weight vector perpendicular to the hyperplane.*

*x is the feature vector of an input instance.*

*b is the bias term.*

The decision rule for classifying a new input instance x can be expressed as:

**f(x) = sign(w · x + b)**

*where sign() is the sign function, assigning a class label based on the positive or negative side of the hyperplane.*

To achieve maximum separation, SVC optimizes the margin, which is the distance between the hyperplane and the nearest data points from both classes. The margin is denoted as 2/‖w‖, where ‖w‖ is the Euclidean norm of the weight vector w.

In the context of email spam classification, SVC can be trained using features extracted from emails, such as word frequencies, presence of specific words, or additional numerical features. The algorithm learns to find the optimal hyperplane that separates spam emails from legitimate emails based on the given features. It takes into account the relationships between the features and their influence on the classification decision.

### 6.2.3 Decision Trees:

Decision trees are non-parametric machine learning models that utilize a tree-like structure for decision-making. They partition the feature space into regions based on feature values and create a tree structure where each internal node represents a decision based on a feature, and each leaf node represents a class label.

*Mathematical Equations:*

The decision-making process in a decision tree involves evaluating feature conditions and making decisions based on those conditions. Each internal node represents a feature and a threshold value, and the branches represent the possible outcomes of the condition.

A decision tree can be represented by a set of if-then rules. The mathematical equations defining the decision tree can vary based on the specific algorithm used for tree construction. One popular algorithm for constructing decision trees is the ID3 (Iterative Dichotomiser 3) algorithm, which uses information gain as the splitting criterion.

Information gain is calculated using the entropy measure, which quantifies the impurity or disorder of a node. The entropy of a node N with respect to a binary class variable C is defined as:

**H(N) = -Σ (p(i) \* log2(p(i)))**

*where p(i) is the probability of class i in node N.*

The information gain of a feature F with respect to a node N is defined as the difference between the entropy of the node before the split (H(N)) and the weighted average of the entropies of the child nodes after the split:

**Gain(F, N) = H(N) - Σ ((|Nv| / |N|) \* H(Nv))**

where Nv is a child node of N resulting from the split based on feature F, |Nv| is the number of instances in Nv, and |N| is the total number of instances in N.

The decision tree learning process continues recursively until a stopping criterion is met, such as reaching a maximum depth or purity of the nodes.

In the context of email spam classification, decision trees can be trained using features extracted from emails, such as word frequencies, presence of specific words, or additional numerical features. The decision tree algorithm learns to create a tree structure that makes decisions based on these features to classify emails as spam or legitimate.

Decision trees have the advantage of being interpretable as they provide a set of if-then rules that explain the decision-making process. They can capture non-linear relationships between features and the class labels and handle both categorical and numerical features.

However, decision trees are prone to overfitting, especially when the tree becomes too complex. Techniques such as pruning and limiting the maximum depth of the tree can help mitigate overfitting.

# 7. Comparative Analysis and Results

## 7.1 Accuracy:

Accuracy measures the overall correctness of the model's predictions and is defined as the ratio of correctly classified instances to the total number of instances. However, in imbalanced datasets where one class dominates the other (as in the case of email spam detection), accuracy alone might not be the most informative metric.

## 7.2 Precision:

Precision is a metric that quantifies the model's ability to correctly identify positive instances (spam emails) among the instances it predicts as positive. It is calculated as the ratio of true positives to the sum of true positives and false positives. Precision focuses on minimizing false positives, which means minimizing the instances incorrectly classified as spam.

## 7.3 Recall (Sensitivity or True Positive Rate):

Recall measures the model's ability to correctly identify positive instances (spam emails) among all actual positive instances. It is calculated as the ratio of true positives to the sum of true positives and false negatives. Recall focuses on minimizing false negatives, which means minimizing the instances incorrectly classified as legitimate emails.

## 7.4 The significance of these parameters in evaluating the models:

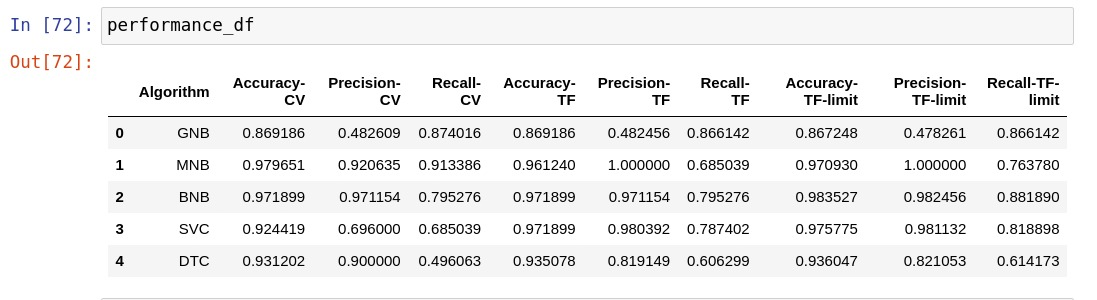
Considering the dataset's small size and the presence of a high proportion of non-spam emails, accuracy alone may not be the most informative metric. Instead, we focus on evaluating models based on two key parameters: precision and recall.

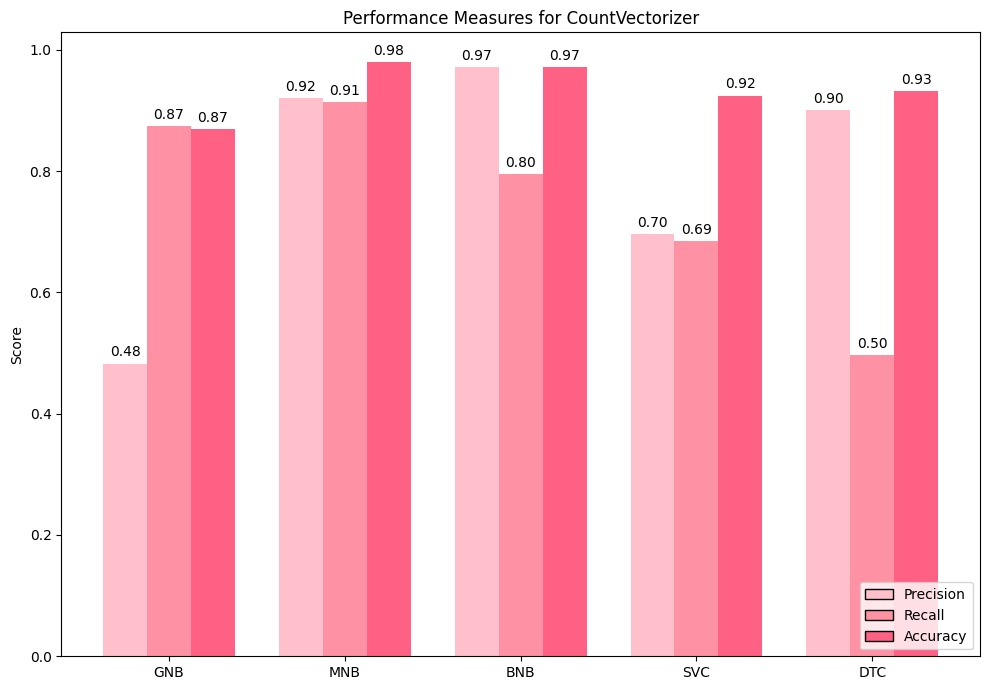
**High Precision:** A high precision indicates that the model has a low false positive rate, meaning that the model correctly identifies most of the predicted positive instances as actual positive instances. In the context of email spam detection, high precision implies that the model is effective in accurately classifying emails as spam, minimizing the chances of legitimate emails being classified as spam.

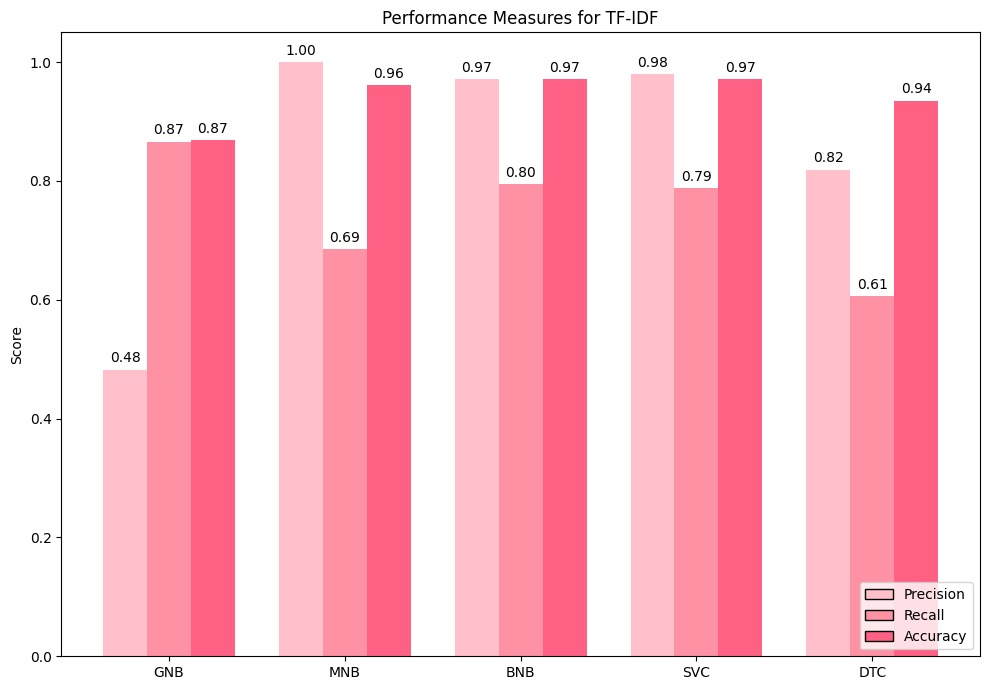
**Significant Recall:** A significant recall indicates that the model has a low false negative rate, meaning that the model correctly identifies most of the actual positive instances (spam emails) as positive. In the context of email spam detection, significant recall implies that the model effectively captures a large proportion of spam emails, minimizing the chances of spam emails being incorrectly classified as legitimate.

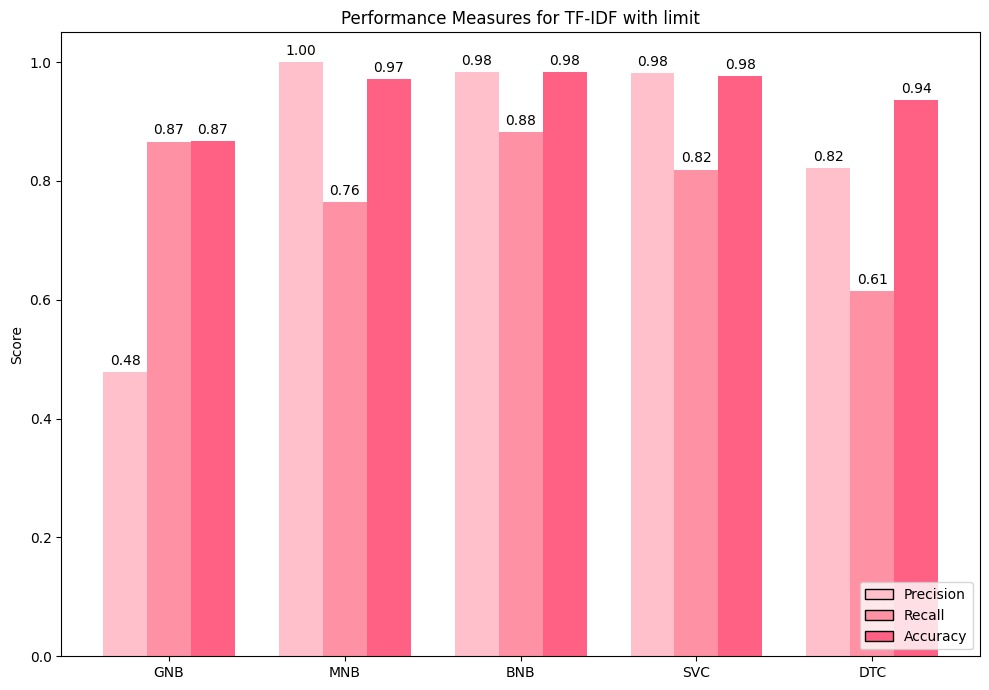
Our goal is to achieve high precision along with significant recall, striking a balance between minimizing false positives and false negatives. By analysing these parameters, we can gain insights into the performance of each model and make informed decisions regarding their suitability for email spam detection.

To provide a comprehensive comparison, tables and graphs were created to showcase the results of each model, highlighting their respective performances on these metrics.



****

****

****

The primary requirement of the email classification model is to minimize the misclassification of non-spam emails as spam. Marking important emails as spam can have significant consequences, leading to missed opportunities and potential loss of crucial communication. Therefore, it is essential for the model to have a low false positive rate or high precision.

Additionally, the model should effectively identify a significant proportion of spam emails to ensure robust spam detection. This requirement is captured by the recall metric, which measures the ability of the model to correctly identify positive instances (spam emails) from the entire population of positive instances.

Considering these requirements, the insights from the comparative analysis of the models can be summarized as follows:

* **Naive Bayes algorithms (GNB, MNB, BNB)** demonstrate a good balance between precision and recall. They achieve high precision values, indicating a low rate of false positives. This means that they are less likely to classify non-spam emails as spam, reducing the risk of important emails being wrongly labelled. These algorithms also exhibit high recall scores, suggesting their ability to identify a large portion of actual spam emails. Therefore, Naive Bayes algorithms fulfil the requirements of minimizing false positives and effectively detecting spam emails.
* **Support Vector Classifier (SVC)** shows competitive performance with high accuracy scores. Although it has slightly lower precision compared to Naive Bayes algorithms, it compensates with excellent recall scores. SVC is capable of capturing a significant proportion of spam emails, ensuring efficient spam detection. However, it is important to carefully consider the tolerance for false positives while using SVC, as it may result in a slightly higher misclassification rate for non-spam emails.
* **Decision Tree Classifier (DTC)** achieves relatively lower precision and recall values compared to the Naive Bayes algorithms. This indicates a higher risk of misclassifying non-spam emails as spam and potentially missing some spam emails. Hence, DTC may not be the ideal choice if the goal is to minimize false positives and capture a high proportion of spam emails.

Considering the requirements specified, the **Naive Bayes algorithms, particularly MNB and BNB**, appear to be the most suitable choices for the email classification model. They strike a balance between minimizing false positives and effectively detecting spam emails.

# 8. CONCLUSION

This project developed an email spam detection model using machine learning algorithms. The Naive Bayes classifiers, particularly MNB and BNB, demonstrated strong performance by achieving high accuracy, precision, and recall values. These models effectively minimized false positives while accurately identifying spam emails. The findings emphasize the importance of selecting suitable algorithms for email spam classification, and the developed models can enhance communication security and efficiency. Further improvements and fine-tuning can be explored for optimizing performance in various scenarios. Overall, this project contributes valuable insights to the field of email spam detection and provides practical solutions for mitigating the challenges of spam emails.

# REFERENCES

1. Data set: <https://archive.ics.uci.edu/ml/datasets/spambase>
2. Fundamentals of Machine Learning for Predictive Data Analytics by John D. Kelleher, Brian Mac Namee, and Aoife D’Arcy.
3. Natural Language Processing with Python" by Steven Bird, Ewan Klein, and Edward Loper.