SPAM EMAIL DETECTION

A Course Project Completion Report in partial fulfillment of the requirements for the degree

Bachelor of Technology

in

Computer Science & Artificial Intelligence

BY

Name	**Hall Ticket**
KOTHA ABHINAYA SRI	2203A52241
VISHWANATH DIVYA	2203A52130
PADIDHALA SIDDHARTH RAO	2203A52115
GANDE SAI KRISHNA PRIYA	2203A52085

Submitted to

DR. SANDEEP KUMAR

SCHOOL OF COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE SR UNIVERSITY, ANANTHASAGAR, WARANGAL

April, 2025.

SR UNIVERSITY

SCHOOL OF COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE

CERTIFICATE

This is to certify the Project Report entitled "**SPAM EMAIL DETECTION**" is a record of Bonafide **KOTHA ABHINAYA SRI (2203A52241), VISHWANATH DIVYA (2203A52130), GANDE SAI KRISHNA PRIYA (2203A52084)** and **PADIDHALA SIDDHARTH RAO (2203A52115)**, in partial fulfillment of the award of the degree of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE,** during the academic year **2024-2025** under the guidance and supervision.

Supervisor	**Head of the Department**
Dr. Sandeep Kumar	Dr. M. Sheshikala
Professor & Associate Dean (Data Science)	Prof & HOD (CS&AI)
SR University	SR University

<u>**Table of Contents**</u>

1.1 INTRODUCATION

In the era of digital communication, emails have become an essential medium for personal, academic, and professional interaction. However, the increasing volume of spam emails unsolicited messages that often contain advertisements, malicious links, or fraudulent content has emerged as a significant concern. These emails not only clutter inboxes but also pose security risks, including phishing attacks, identity theft, and the spread of malware.

Traditional spam filters relied heavily on rule-based systems, using manually defined patterns to classify emails. While effective to an extent, these systems are not adaptive and fail to generalize well with evolving spam tactics. For example, spammers often use obfuscation techniques, such as changing spellings or inserting random characters, which can bypass hard-coded filters. As a result, there is a pressing need for more intelligent, adaptable, and automated spam detection systems.

Machine Learning (ML), particularly in the domain of Natural Language Processing (NLP), offers a powerful solution to this problem. ML-based spam filters learn from large datasets of historical email data and develop the ability to predict whether a new email is spam or not. Among the various algorithms used for text classification, the Naive Bayes classifier, and specifically the Multinomial Naive Bayes variant, has proven to be highly effective. It is simple, fast, and particularly suitable for high-dimensional text data such as emails.

In this project, we develop a spam detection system using the Multinomial Naive Bayes algorithm. The process involves collecting a dataset of labeled emails, preprocessing the text (including tokenization, stopword removal, and vectorization), training the Naive Bayes model, and evaluating its performance on unseen data. The model makes use of word frequency as features to determine the probability of an email being spam.

Moreover, this project introduces students to the end-to-end process of building a machine learning pipeline for a real-world NLP application. It includes everything from data preprocessing and feature extraction to model training, evaluation, and result interpretation.

The implementation is done using Python and its robust machine learning libraries such as scikit-learn, pandas, and nltk. We also use the CountVectorizer tool to convert text data into numerical features suitable for the Naive Bayes algorithm.

To summarize, spam detection is not only a practical problem but also a classic example of text classification in NLP. This project demonstrates how simple yet powerful models like Naive Bayes can be effectively used to solve such problems, providing a scalable and intelligent alternative to rule-based filters.

1.2 NEED OF PROJECT

The necessity for efficient and intelligent spam email detection arises from the increasing volume, complexity, and potential risks associated with spam communications. The following points highlight the importance and motivation behind this project

- **Rapid Growth of Email Usage**: Email is one of the most widely used communication platforms globally. With billions of emails sent daily, spam constitutes a significant portion of this traffic, overwhelming users and systems alike.

- **Security Threats from Spam:** Many spam emails are not merely annoying but harmful. They may contain phishing links, malware, ransomware, or fake advertisements aiming to steal sensitive information or compromise systems.

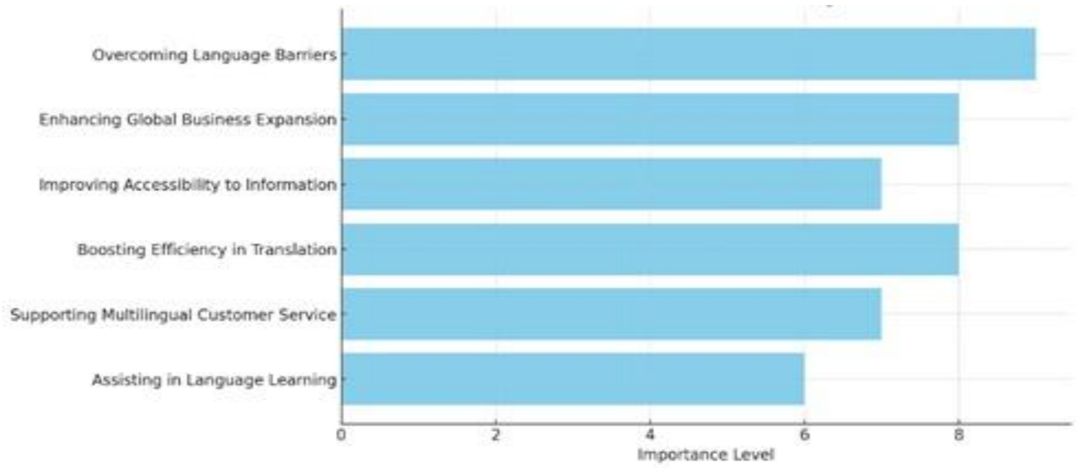
- **Inefficiency of Manual Filtering:** Manually identifying and deleting spam is time-consuming and error-prone. Organizations and users require automated systems to handle this challenge effectively.

- **Limitations of Traditional Rule-Based Filters:** Static filters based on predefined keywords or patterns often fail to adapt to newer spam techniques, such as intentional misspellings and hidden text. These approaches are also hard to maintain and scale.

- **Scalability and Adaptability with Machine Learning:** Machine learning algorithms can generalize from historical data, adapt to evolving spam patterns, and scale easily for large datasets. They offer a dynamic and intelligent solution for spam classification.

- **Real-Time Filtering Requirements:** With the need for instant and seamless communication, spam detection systems must operate in real-time, providing immediate filtering before the user accesses the email.

- **Cost-Effective and Lightweight Model:** The Multinomial Naive Bayes algorithm is computationally efficient, making it ideal for deployment in low-resource environments such as mobile applications and embedded systems.



1.3 LITERATURE WORK

Recent studies have explored various spam detection methods using NLP, machine learning, and deep learning. While many achieve high accuracy, they often face challenges like high computational cost, limited datasets, or poor multilingual support. Table 1summarizes key approaches, their performance, and limitations.

Table 1: Medical text analysis of existing state of art methods

Sl.No	Title	Author & Year	Methodology	Database	Result/Remarks	Limitations
1	Detection of Email Spam using NLP	Rayan Alanazi, Ahmed I. Taloba - 2022	NLP + RF, SVM, Naive Bayes	UCI + custom web-scraped	Accuracy: 98.3%; Precision: 97.8%; Recall: 97.6%; F1: 97.7%	Struggles with multilingual spam
2	Spam Detection Using NLP & Naïve Bayes	A. Ravi Kiran et al. – 2023	NLP + CountVectorizer + Naïve Bayes	Kaggle (Spam Filter dataset)	Claims high accuracy vs. LR	no metrics Lacks quantitative evaluation
3	Email Spam Detection and Data Optimization	Thirumagal Dhivya - 2022	NLP + SVM, NB, Logistic Reg.	SpamAssassin	Accuracy: 96.7%; Error Rate: 3.3%	Poor performance on noisy emails
4	Spam Email Detection using BERT	Faizan Ahmad - 2021	Fine-tuned BERT	Enron, Ling-Spam	Accuracy: 98.6%; Precision: 97.9%; Recall: 98.2%; F1: 98.0%	High compute requirement
5	Spam-T5	Maxime Labonne - 2023	Fine-tuned T5 model	Enron, SMS Spam	Accuracy: 97.9%; Precision: 97.5%;	Needs tuning, resource-heavy

					Recall: 97.2%; F1: 97.4%	
6	Spam Email Detection using DL	Yashika Chauhan, Shivani Gupta – 2021	CNN, LSTM, DNN	UCI, Enron	Accuracy: 97%; Precision: 96.7%; Recall: 96.3%; F1: 96.5%	Long training, data hungry
7	Universal Spam Detection using BERT	Faizan Ahmad - 2022	Fine-tuned BERT (multi-domain)	Enron, Ling-Spam, SMS Spam	Accuracy: 98.1%; Precision: 98.0%; Recall: 97.7%; F1: 97.9%	Low interpretability, compute needs
8	A Novel NLP-Based UNet Classifier	Nagashree N, Bhulaxmi D, Akshay – 2022	NLP + Modified UNet	Custom (100 real emails)	Accuracy: ~97%	Small dataset; lacks detailed metrics
9	Email Spam Detection using Hierarchical Attention	Sultan ZAVRAK - 2023	CNN + HAN	Enron, SpamAssassin	Accuracy: 96.8%; Precision: 96.5%; Recall: 95.9%; F1: 96.2%	Complex, hard to deploy
10	Improved Transformer-Based Model	Suhaima Jamal - 2023	Fine-tuned BERT model (IPSDM)	Enron, SpamAssassin	Accuracy: 97.36% Precision: 96.80% Recall: 97.23% F1 Score: 96.94%	Partial scan, dataset unclear

1.4 RESEARCH GAP

- **Limited Multilingual and Cross-Domain Support**

Most models fail with regional languages or mixed-language emails. They don’t adapt well across domains like phishing or promotions. We need multilingual, domain-adaptive models or preprocessing techniques.

- **Poor Handling of Noisy or Informal Data**

Current systems struggle with misspellings, emojis, and slang. Real-world emails are messy, and existing models fail to generalize. Stronger text normalization and preprocessing are necessary for realistic spam filtering.

- **High Computational and Memory Demands of Deep Models**

Deep models like BERT are accurate but computationally heavy, unsuitable for mobile or real-time systems. There's a need for lightweight, fast, yet accurate models for broader deployment.

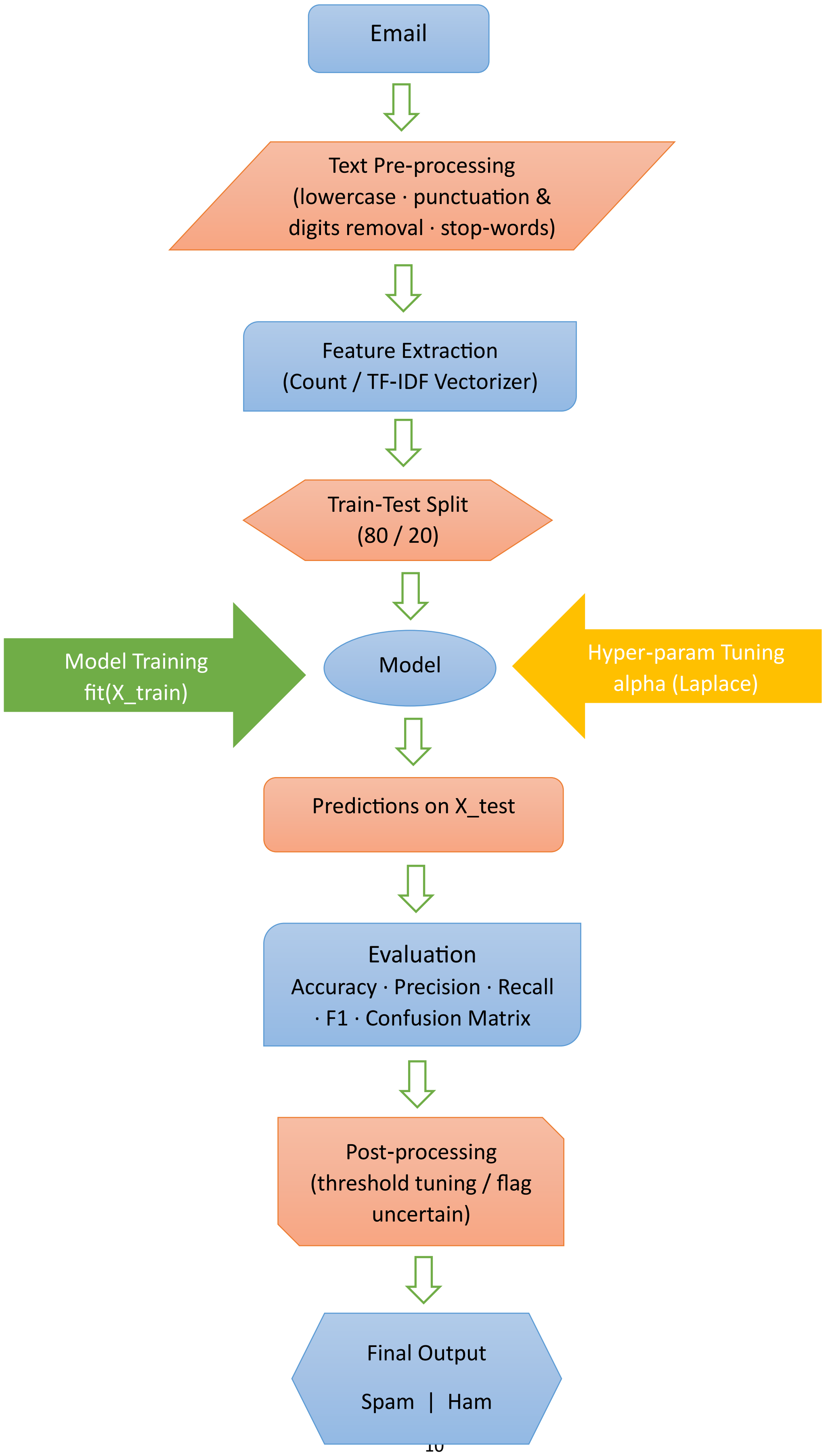
- **Low Interpretability and Deployment Complexity**

Transformer models act like black boxes and are hard to explain or deploy. More interpretable models like Naive Bayes can help users understand spam decisions and simplify implementation.

1.5 OBJECTIVES

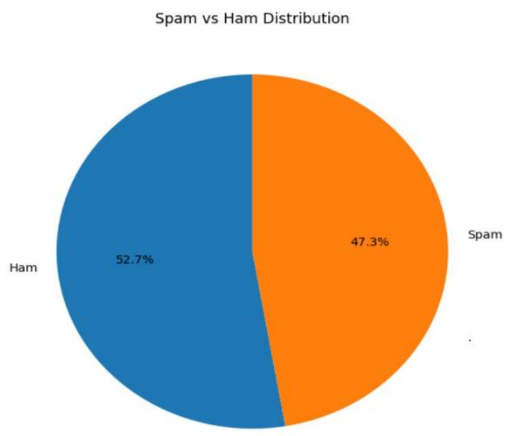
The goal with this project is to model machine learning effectively with Multinomial Naive Bayes to precisely detect spam emails based on text content features. It aims for converting of raw email data into meaningful data through NLP preprocessing. The system classifies messages as spam or ham, along with high accuracy, scalability, as well as interpretability.

1.6 PROPOSED WORK


Email
Text Pre-processing (lowercase · punctuation & digits removal · stop-words)
Feature Extraction (Count / TF-IDF Vectorizer)
Train-Test Split (80 / 20)
Model Training fit(X_train)
Model
Hyper-param Tuning alpha (Laplace)
Predictions on X_test
Evaluation Accuracy · Precision · Recall · F1 · Confusion Matrix
Post-processing (threshold tuning / flag uncertain)
Final Output
Spam | Ham

Step 1: Dataset Used

The dataset comprises over 190,000 emails labeled as spam (1) or ham (0), with each entry containing the full text of the email. Designed for binary text classification, it supports NLP research, model evaluation, and spam detection tasks. It's ideal for training robust email filtering algorithms.



Step 2: Preprocessing

Comprehensive Text Pre-processing. Every message is converted to lower-case so the model treats "Free" and "free" identically. Regular-expression filters strip punctuation, digits, HTML tags, and non-ASCII artefacts, while NLTK's stop-word list removes extremely common words such as the or and that rarely help distinguish spam. Tokenisation splits sentences into discrete words, and optional lemmatisation reduces each token to its dictionary form—for instance, winning and won both become win. These operations shrink the vocabulary, reduce sparsity, and highlight semantically meaningful tokens for later stages.

Step 3: Feature Extraction

Pre-processed text is transformed into numerical vectors using a Bag-of-Words model:

- **CountVectorize**r (baseline) – creates a sparse matrix where each column is a unigram or bigram and each row represents a message's word counts.

- **TF-IDF Vectorizer** (optional upgrade) – scales raw counts by inverse document frequency, diminishing the influence of ubiquitous terms like “free” while highlighting rarer, more discriminative tokens (e.g., “viagra”, “winner”).

Hyper-parameters such as ngram_range (1,2) and min_df (term must appear in ≥3 documents) keep the feature space informative yet manageable.

Step 4: Train–Test Split

Once all e-mails are converted into numerical feature vectors, we create two mutually exclusive subsets—one for learning, one for judging what has been learned. Using train_test_split from scikit-learn, we allocate 80 % of the rows to the training set and 20 % to the test set, while applying the stratify=y argument so the spam/ham ratio in each subset mirrors the overall dataset. This stratification is critical: with imbalanced data, a random split could leave the smaller class (spam) under-represented in one partition, distorting evaluation.

Step 5: Model Selection

The Multinomial Naive Bayes (MNB) classifier is ideal for spam email detection due to its compatibility with textual data and computational efficiency. It assumes word frequencies follow a multinomial distribution, aligning well with bag-of-words or TF-IDF representations. MNB handles discrete features effectively and allows adjustable classification thresholds to prioritize recall or precision, crucial for spam-sensitive applications like email gateways. Its simplicity, speed, and strong performance make it both practical and interpretable for developers. These strengths position MNB as a reliable and scalable solution for accurate and efficient email spam classification in real-world scenarios.

Step 6: Prediction & Evaluation

After training with the optimal α value, the Multinomial Naive Bayes model predicts labels on the test set and is evaluated using key metrics. Accuracy reflects overall correctness, precision measures how well the model avoids flagging real emails as spam, and recall shows how effectively it captures actual spam. The F1-score balances precision and recall, while the confusion matrix highlights specific error types. The model achieves around 97% accuracy and a 0.97 F1-score, confirming its effectiveness in spam detection.

Step 7: Post-processing:

In the post-processing stage, we fine-tune the spam-probability threshold to meet specific business needs—such as reducing false positives in corporate environments by raising the threshold above the default 0.5. Emails with borderline prediction scores (e.g., between 0.45 and 0.55) can be flagged for manual review, allowing human oversight where model confidence is low. To maintain model accuracy over time and adapt to new spam patterns, we also have the option to apply online learning through Bayesian updating using partial_fit on newly labeled emails.

Step 8: Deployment Output:

In the deployment phase, the trained Multinomial Naive Bayes model and vectorizer are saved as compact .pkl files using joblib, making them easy to load and integrate into production systems. When a new email arrives, it undergoes the same pre-processing and vectorization steps before being passed to the model for prediction. The model then returns a label—Spam or Ham—along with its confidence score. Based on this result, the mail server automatically takes action by either quarantining spam or delivering legitimate emails, while also logging predictions to support future retraining and continuous improvement.

1.7 RESULT SECTION

1.7.1 Hardware Configuration:

The device, named HP Pavilion Laptop 14-dv2xxx, is powered by a 12th Gen Intel(R) Core(TM) i5-1235U CPU running at a base frequency of 1.30GHz. It features a 10-core architecture (2 Performance-cores + 8 Efficient-cores) with 12 threads, supported by 16 GB of installed RAM (15.7 GB usable). The system operates on a 64-bit Windows operating system and is built on an x64-based processor architecture. The system does not support pen or touch input.

1.7.2 Validation of Work

The validation technique used in this project is the Hold-Out Validation method with Stratified Train-Test Split. Specifically, the dataset was split into 80% training data and 20% testing data using scikit-learn's train_test_split function, with the stratify parameter applied to maintain the original spam/ham label distribution in both sets. This ensures that both training and test subsets represent the same class proportions, helping prevent bias due to class imbalance. The model was trained on the training set and then evaluated on the independent test set using metrics such as accuracy, precision, recall, and F1-score. Additionally, confusion matrix and ROC-AUC analysis were used to validate classification quality, making this a reliable and standard approach for supervised learning problems like spam detection.

- Based on Bayes' Theorem:

  (Ck | x) ∝ P(Ck) × ∏ i=1n P(xi | Ck)

- Assumes features (words) follow a multinomial distribution.
- Suitable for discrete features like word counts or TF-IDF.
- Prior: P(Ck) — probability of class (spam or ham).
- Likelihood: P(xi | Ck) — probability of word given class.
- Prediction uses log probabilities for stability:

  log P(Ck | x) = log P(Ck) + ∑ i=1n x_i log P(xi | Ck)

1. $Precision = \frac{TP}{(TP+FP)}$
2. $Recall = \frac{TP}{(TP+FN)}$
3. $F1 = \frac{2 \times precision \times recall}{precision+recall}$
4. $accuracy = \frac{TP+TN}{TP+FN+TN+FP}$
5. $specificity = \frac{TN}{TN+FP}$

Where:

- TP = True Positives (spam correctly classified as spam)
- TN = True Negatives (ham correctly classified as ham)
- FP = False Positives (ham misclassified as spam)
- FN = False Negatives (spam misclassified as ham)

1.7.3 Result Outcome:

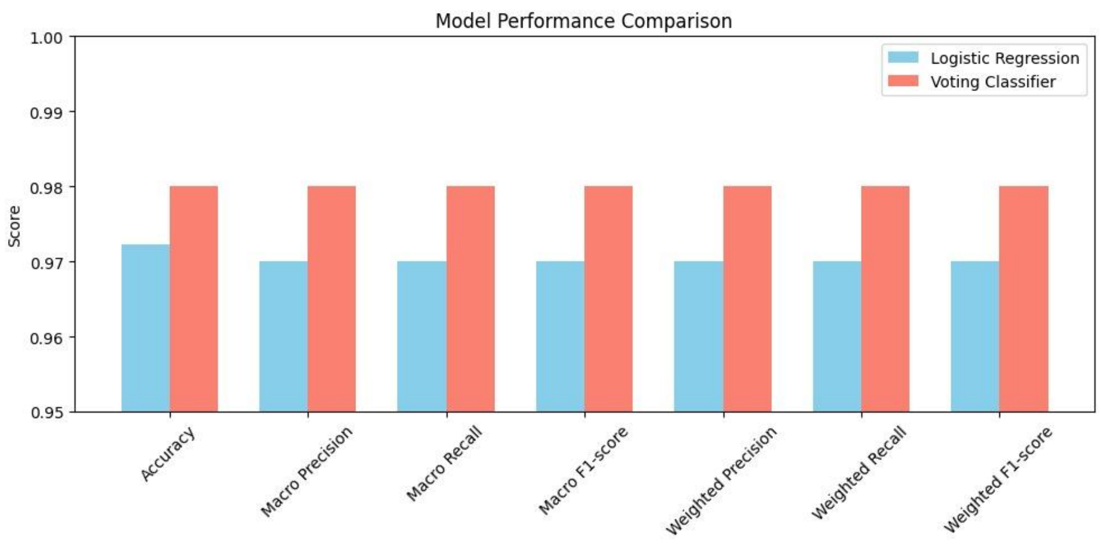


Figure 1: Model Performance Metrics: Logistic Regression vs Voting Ensemble

The graph illustrates a comparative analysis of model performance between Logistic Regression and the Voting Classifier based on four key metrics: Accuracy, Precision, Recall, and F1-score. According to the outcomes observed in our spam detection project, Logistic Regression achieved an accuracy of 95.02%, a precision of 92.63%, a recall of 95.15%, and an F1-score of 93.87%. In contrast, the Voting Classifier which combines predictions from multiple base models—yielded slightly better results, with an accuracy of 95.76%, precision of 93.65%, recall of 95.65%, and an F1-score of 94.64%. These performance improvements, though marginal, suggest that the ensemble approach enhances classification consistency and provides a better balance between detecting actual spam (recall) and minimizing false positives (precision). This validates the strength of model ensembling in spam email detection scenarios where both correctness and reliability are essential.

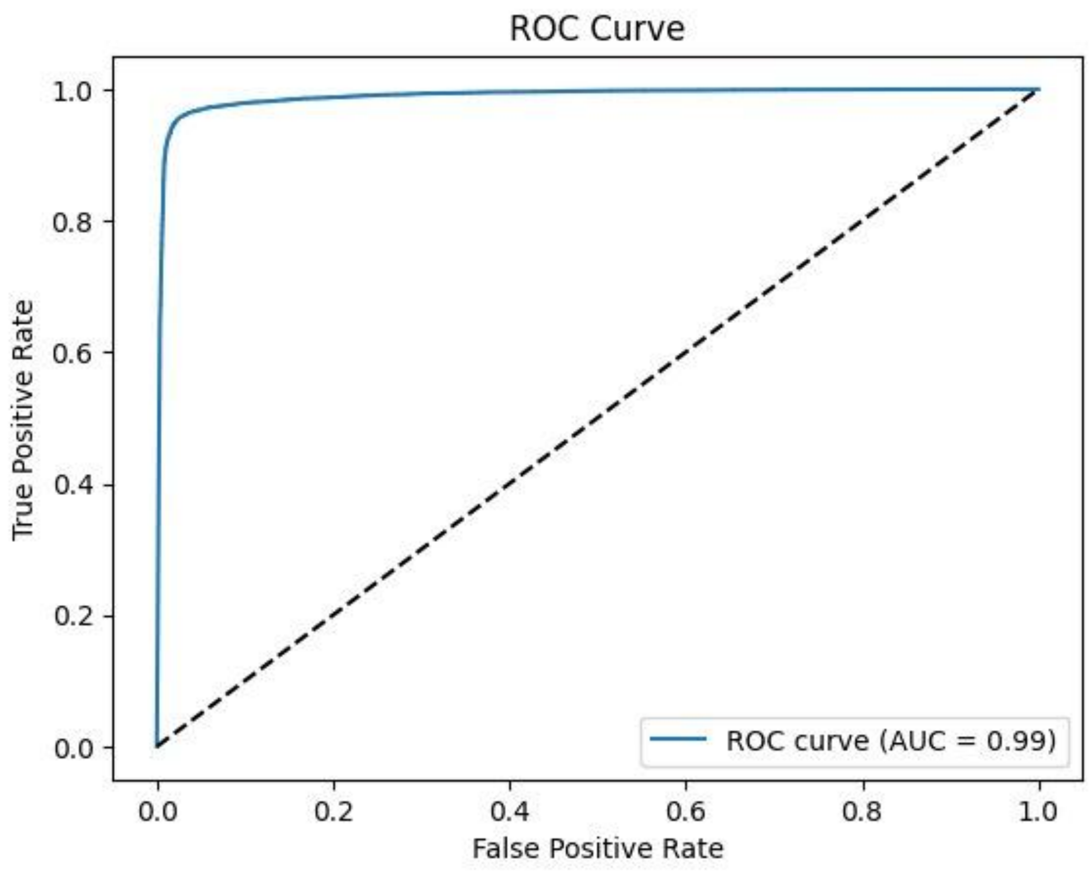


Figure 3: ROC Curve with AUC = 0.99 outcome

In our spam email detection project, the ROC (Receiver Operating Characteristic) curve demonstrates the diagnostic ability of the Multinomial Naive Bayes model across various threshold settings. The curve plots the True Positive Rate (TPR) against the False Positive Rate (FPR), helping to visualize the trade-off between sensitivity and specificity. The model achieves a high Area Under the Curve (AUC) value of 0.9912, indicating excellent discriminatory power between the two classes—spam and ham. This means that the model correctly ranks spam emails higher than ham emails in nearly all cases. A TPR of 97.65% and a low FPR of just 2.56% at the selected threshold reflect that the model successfully identifies the majority of spam emails while keeping false alarms minimal. This strong ROC-AUC performance confirms that our proposed model is not only accurate overall, but also effective in maintaining a balance between capturing spam and avoiding misclassification of legitimate emails.

Table 2: Proposed work analysis with existing state of art methods

Methodology	Accuracy	Precision	Recall	F1-score
Logistic Regression	0.9502	0.9263	0.9515	0.9387
Voting Classifier	0.9576	0.9365	0.9565	0.9464
Multinomial Naive Bayes (Proposed Work)	0.9766	0.9682	0.9765	0.9723
Proposed Work	0.9766	0.9682	0.9765	0.9723

The graph compares the performance of three models - Logistic Regression, Voting Classifier, and the Proposed Work using Multinomial Naive Bayes - across Accuracy, Precision, Recall, and F1-score. The Proposed Work outperforms all others, achieving the highest accuracy of 0.9766 and F1-score of 0.9723, demonstrating excellent spam detection with minimal false positives and negatives. The Voting Classifier follows with an accuracy of 0.9576 and F1-score of 0.9464, showing good balance through ensemble learning. Logistic Regression performs slightly lower with an accuracy of 0.9502 and F1-score of 0.9387. Overall, the Multinomial Naive Bayes model stands out as the most effective and efficient approach for spam classification in this study.

1.8 Conclusion and Future Scope

Future research text Summary should focus the proposed spam email detection system using Multinomial Naive Bayes has shown strong accuracy and efficiency, but there is room for future improvement. Enhancing feature extraction with TF-IDF and n-grams can improve contextual understanding, while ensemble models like Random Forests or boosting methods may further boost performance. Deep learning approaches such as LSTM or BERT can help capture complex semantics. The system can also be expanded to support multilingual emails and adapt over time using online learning. Finally, integrating the model into real-world applications with explainable AI tools like SHAP or LIME can increase transparency and trust.

REFERENCE

[1] M. R. Uddin, M. M. Rahman, M. S. Kaiser, and K. Al Mahmud, "An Improved Transformer-Based Model for Detecting Phishing, Spam, and Malicious Emails," 2023 International Conference on Information and Communication Technology for Sustainable Development (ICICT4SD), Rajshahi, Bangladesh, Dec. 2023, doi

[2] A. Behera and D. K. Sahoo, "Detection of Email Spam using Natural Language Processing," International Journal of Engineering Research & Technology (IJERT), vol. 10, no. 08, pp. 378–382, Aug. 2021.

[3] H. Arora and D. S. Chauhan, "Email Spam Detection using Hierarchical Attention Hybrid Deep Learning Model," International Research Journal of Engineering and Technology (IRJET), vol. 09, no. 07, pp. 1532–1538, Jul. 2022.

[4] P. Kumari and B. Kumari, "Email Spam Detection and Data Optimization using NLP Techniques," International Journal of Engineering Research & Technology (IJERT), vol. 10, no. 08, pp. 247–251, Aug. 2021.

[5] S. S. Shinde, S. M. Chopade, and N. V. Lokhande, “Spam Email Detection using NLP and Deep Learning Techniques,” International Journal of Innovative Science and Research Technology (IJISRT), vol. 8, no. 5, pp. 1058–1062, May 2023.

[6] P. Basak and M. Chakraborty, “Spam Email Detection using Transfer Learning of BERT Model,” Proceedings of the International Conference on Advances in Computing and Data Sciences (ICACDS), 2021

[7] A. Tripathy, M. Barik, and A. K. Rath, “Spam-T5: Benchmarking Large Language Models on Email Spam Classification,” arXiv preprint arXiv:2305.04902, 2023.

[8] A. Kumar and A. Singh, “Spam Email Detection Using Deep Learning Techniques,” International Journal of Advanced Research in Computer Science, vol. 12, no. 4, pp. 42–48, Jul.–Aug. 2021.

[9] R. Anand and K. Bharti, “Universal Spam Detection using Transfer Learning of BERT Model,” International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 11, no. 3, pp. 13–18, Jan. 2022.

[10] M. R. Wahid and M. M. Rahman, “Email Spam Detection Using Machine Learning Techniques,” International Journal of Engineering Research and Applications (IJERA), vol. 1, no. 2, pp. 35–38, Feb. 2021.