**Enhanced Spam Classification with Improved Accuracy Using Refined Stacking with Voting Meta - Classifier**

Sathvik Eppakayala1, Kanchi Nissy2, Dr. Popuri Srinivas Rao3

1, 2 Department of Computer Science and Engineering (Data Science), R.V.R & J.C. College of Engineering, Guntur, AP, India.

3Assoc. Prof., Department of Computer Science and Engineering (Data Science), R.V.R & J.C. College of Engineering, Guntur, AP, India.  
Address

1eppakayalasathvik72@gmail.com

2nissykanchi@gmail.com

3srinivasaraopopuri@rvrjc.ac.in

Abstract— Spam email classification is a significant cybersecurity problem, necessitating sophisticated classification approaches to address emerging threats. In this research, our objective is an advanced voting ensemble model that combines several machine learning classifiers—Logistic Regression, Decision Tree, and Multinomial Naive Bayes,—to achieve optimal classification precision. As compared to conventional single-classifier models, our method utilizes a Voting Classifier as the final estimator in the stacking model, which greatly enhances prediction capability and prediction accuracy. By intense preprocessing, such as removal of stopwords, and TF-IDF vectorization, our model is able to grasp significant text patterns. Experimental results show a highest-ever accuracy of 98.53% and a highest-ever recall measure of 99.09%, outperforming other spam detection algorithms. These results identify the power of ensemble learning, specifically stacking with a Voting Classifier as Meta classifier, as a robust algorithm to improve email security and text categorization tasks.

Keywords— *Machine Learning, Ensemble Methods, Stacking Classifier, Voting Classifier, Refined machine learning*

**1. INTRODUCTION**

Email communication has emerged as a necessary aspect of contemporary digital communication, playing a vital role as a fundamental method for personal, business and professional communication. Despite its application, the present problem of spam mails, which dominate a great majority of mail traffic worldwide. Spam emails consist of unwanted messages, phishing scams, and malicious information, which represent cyber frauds like financial fraud, data breaches, and identity thefts. Traditional spam detection methods, including rule-based filters, blacklist techniques, struggle to keep up with evolving spam tactics, necessitating more advanced machine learning-based solutions.

Machine Learning (ML) has emerged as a powerful approach for spam classification, leveraging algorithms that learn from past email data to identify spam trends and patterns. While there has been usage of individual classifiers like Logistic Regression, Decision Tree Classifier, and Naïve Bayes, individually they suffer with certain limitations leading to reduced accuracy and recall scores. To Overcome this issue, we have proposed this system that average several classifiers and several NLP techniques in order to provide better prediction quality and accuracy. Our system involves with stacking ensemble model and voting ensemble model to provide promising solution by attaining the best classification using VotingClassifier and then it is passed as final\_estimator to the Stacking Ensemble model to achieve higher classification accuracy.

This Experiment involves an Enhanced stacking model, using Logistic Regression, Decision Tree, and Multinomial Naïve Bayes as the base estimators and a Voting Classifier as a meta estimator. This new method attains an accuracy of 98.53% and recall score of 99.09%, surpassing the conventional machine learning models. Finally, the model is deployed using pickle library in python and used for real time classification.

The rest of the paper is structured as follows: [Section 2](#sec2) presents Literature Survey. [Section 3](#sec3) explains the Scope and Objectives. [Section 4](#sec4) explains the datasets, preprocessing methods, and model choice. [Section 6](#fig6) outlines our proposed approach, encompassing the stacking framework and voting classifier as meta classifier. [Section 9](#fig17) reports the experimental results and evaluation metrics. [Section 10](#secend) conclude the research with observations on future improvements and possible research directions

**2. LITERATURE SURVEY**

Spam email classification is a vital field of research in machine learning and cybersecurity. Different methods have been used to enhance accuracy, precision, and recall in the detection and filtering of spam emails. Machine learning, and ensemble methods have been used by the researchers to enhance the detection of spam. Common traditional classifiers such as Naïve Bayes, Decision Trees, and Support Vector Machines have been extensively utilized, yet ensemble models such as Stacking, Boosting, and Voting classifiers have demonstrated major performance enhancements.

Ensemble learning methods, which combine multiple classifiers to make a final prediction, have shown remarkable improvements in spam detection accuracy. Recent studies highlight the effectiveness of stacking and voting ensemble models in enhancing classification performance.

* [**Sahoo et al.** (2024)](#r1) introduced an ensemble method for spam classification, demonstrating that combining classifiers like **Logistic Regression, Decision Trees, and Random Forest** leads to better spam detection rates. Their method improved classification accuracy compared to single classifiers.
* [**Hoque et al.** (2024)](#r2) developed an **optimized soft voting ensemble classifier** that integrates multiple machine learning algorithms. Their approach outperformed traditional classifiers by leveraging an optimized weighting system for ensemble decisions.
* [**Alzahrani (2024)**](#r2) proposed an **explainable AI-based framework** using **stacking classifiers** to enhance spam detection. Their work emphasized the need for transparency in AI-driven spam filters, providing interpretability alongside high accuracy.

Stacking classifiers have gained popularity as they leverage predictions from multiple base classifiers before passing them to a meta-classifier for final decision-making.

* [**Kamalesh et al.** (2024)](#r3) introduced an **enhanced spam classification model** that integrates text preprocessing techniques with deep learning and ensemble models. They reported significant improvements in classification performance using stacking models.
* [**Rahman et al.** (2024)](#r4) conducted a **comparative analysis of hybrid voting ensembles and Bi-LSTM networks** for SMS spam classification. Their study found that ensemble techniques consistently outperformed deep learning models in traditional text-based spam detection.

Voting classifiers, which combine predictions from multiple classifiers using majority or weighted voting, have been found effective in spam detection.

* [**Mankar et al.** (2024)](#r3) analysed **spam SMS detection using ensemble methods** and found that a voting classifier integrating **Decision Trees and Random Forest** significantly enhanced accuracy compared to individual classifiers.
* [**Sharma et al.** (2024)](#r3) designed a **hybrid classifier for spam detection**, combining **Naïve Bayes, SVM, and ensemble techniques**. Their approach increased classification accuracy and reduced false positives in email spam detection.
* [**Tyagi et al.** (2024)](#r5) explored **SMS spam classification with ensemble learning** and concluded that machine learning-based classifiers were more effective than rule-based spam filters, particularly when using an ensemble voting system.

**3. OBJECTIVES & SCOPE**

Our work aims to the following objectives:

* To create a sophisticated spam classification model based on ensemble learning methods, namely stacking and voting classifiers, to improve email security.
* To compare the performance of single classifiers like Logistic Regression, Decision Tree, and Multinomial Naïve Bayes and tune their combination for greater classification accuracy.
* To reduce the computational speed of the model training & testing.
* To enhance spam detection performance through ensemble learning methods for improved precision, recall, and F1-score over standard single-model classifiers.
* To apply effective text preprocessing methods and NLP methods, such as stopword elimination, special character removal, URL’s elimination, tokenization, and TF-IDF vectorization, to enhance feature extraction with features of 20000 and ngram\_range is of trigrams, and model quality.
* To test the designed model with real-world email datasets to ensure robustness and generalizability across different spam detection tasks.
* To improve interpretability of spam classification models through the analysis of the contribution of various classifiers within the ensemble and their optimization in weights for optimal decision-making.
* In order to help the domain of cybersecurity through offering an efficient and scalable model for spam detection that can be used within current email filtering systems.

The Scope of our works involves in

* The study aims to improve email spam filtering by using ensemble learning techniques to enhance classification accuracy
* Advanced text preprocessing methods like stopwords removal, and TF-IDF vectorization are applied to improve spam detection efficiency.
* The system is optimized to handle large email datasets, ensuring high performance in real-world applications.

**4. DATA PREPROCESSING METHODOLOGY**

**4.1. Datasets**

We used the [Spam Apache](https://github.com/sathvikeppakayala/Spam-Classification.git), [Spam Enron](https://github.com/sathvikeppakayala/Spam-Classification.git) and spam\_ham\_data datasets, which together comprised about 62,000 emails. For improving the diversity and strength of the dataset, we also incorporated other datasets, thus making it a total of 62,111 real-world emails. This large dataset helps in building a more generalized and effective spam classification model.

**4.2. Data Cleaning**

For the structure and organization of the dataset, some steps of cleaning data were taken. The labels for spam were first standardized by replacing all instances of "-1" (spam) as "spam" and others as "ham" for maintaining uniformity in datasets. Second, all unnamed or irrelevant columns were dropped in order to remove useless data. In order to preserve data integrity, missing or null email entries were detected and discarded, avoiding misclassification caused by missing information. These cleaning steps made the dataset properly prepared to be further processed and used with machine learning algorithms.

**4.3. Dataset Balancing**

Spam databases tend to be imbalanced, with much lower spam emails than non-spam (ham) emails, and this tends to produce biased classification models. To counter this, random undersampling was used to even out the database. Ham emails were decreased to fit into the number of spam emails so that both classes had an equal representation. This method avoids the classifier from becoming biased towards the majority class and enhances the performance of the overall model by ensuring that spam emails are properly classified.

**4.4. Removal of URL’s, Special Characters**

In this work, we have used nltk text cleaning methods to remove URL’s and Special Character, this improves model accuracy and performance. This approach helped the work, to identify the spam mails without URL’s and Special Characters.

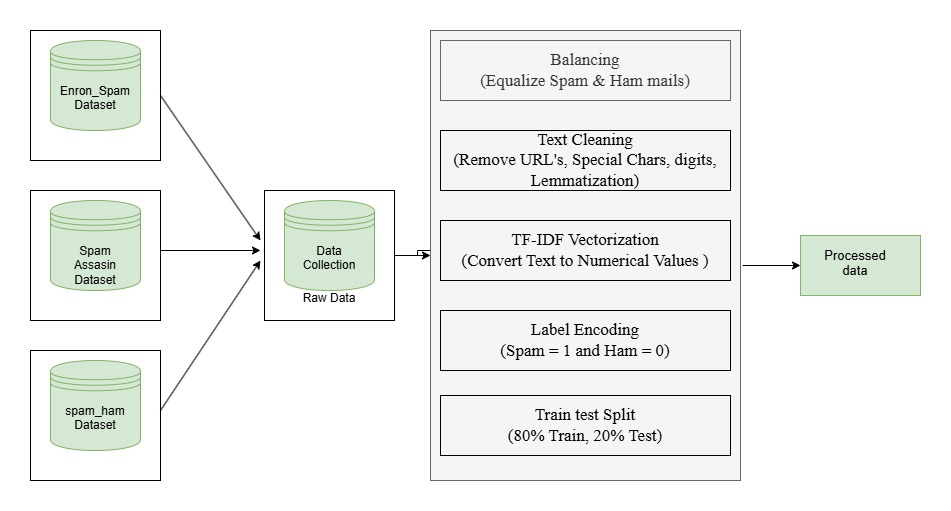
**4.4. Text Preprocessing**

Figure 1. Data Preprocessing Architecture

In order to improve the quality of text data for machine learning, some text preprocessing methods were used. First, all the content of emails was converted into lowercase to maintain consistency in text representation. Second, stopword removal was done to remove frequent words like "the," "is," and "and," which do not carry much meaning for spam classification. Tokenization was then used to split the text into separate words or tokens. Lastly, TF-IDF (Term Frequency-Inverse Document Frequency) vectorization was applied to transform the pre-processed text into numerical feature representations, and the top 20000 most significant features were extracted and used ngram\_range of (1, 3). These preprocessing operations assisted in the enhancement of the spam classifying model's efficiency and accuracy by targeting the most significant words within the dataset.

**5. SOFTWARE & HARDWARE REQUIREMENTS**

To implement the Intelligent Spam Classification Using Ensemble Stacking & Voting Method, the following software and hardware requirements must be met to ensure smooth execution and optimal performance.

**Software Requirements**

1. **Operating System**:
   * Windows 10/11, Linux (Ubuntu 18.04 or later), macOS
2. **Programming Language**:
   * Python 3.8 or later
3. **Libraries and Frameworks**:
   * **Machine Learning**:
     + scikit-learn (for classification models)
     + numpy, pandas (for data handling)
   * **Text Processing**:
     + nltk (for stopword removal, tokenization)
     + scikit-learn (TF-IDF vectorization)
4. **Development Environment**:
   * Jupyter Notebook / VS Code / PyCharm / Google Collab

**Hardware Requirements**

1. **Processor**:
   * Minimum: Intel Core i5 (or equivalent AMD Ryzen 5)
   * Recommended: Intel Core i7/i9 or AMD Ryzen 7/9 (for faster model training)
2. **RAM**:
   * Minimum: 8GB (for small datasets)
   * Recommended: 16GB or more (for handling large datasets efficiently)
3. **Storage**:
   * Minimum: 256GB SSD (for smooth execution)
   * Recommended: 512GB SSD or more (for handling multiple datasets)

**6. PROPOSED SYSTEM**

**6.1. Proposed system Flowchart**

The [Figure 2](#fig2) represents the proposed architecture for the system proposed in this work

**6.2. Methodology**

* Data Collection and Preprocessing (Explained in [section 4](#sec4))
* Training and testing samples
* Building model
* Model Evaluation
* Web interface

**6.3. Training and testing samples**

The preprocessed data was split into two subsets to facilitate proper training and evaluation:

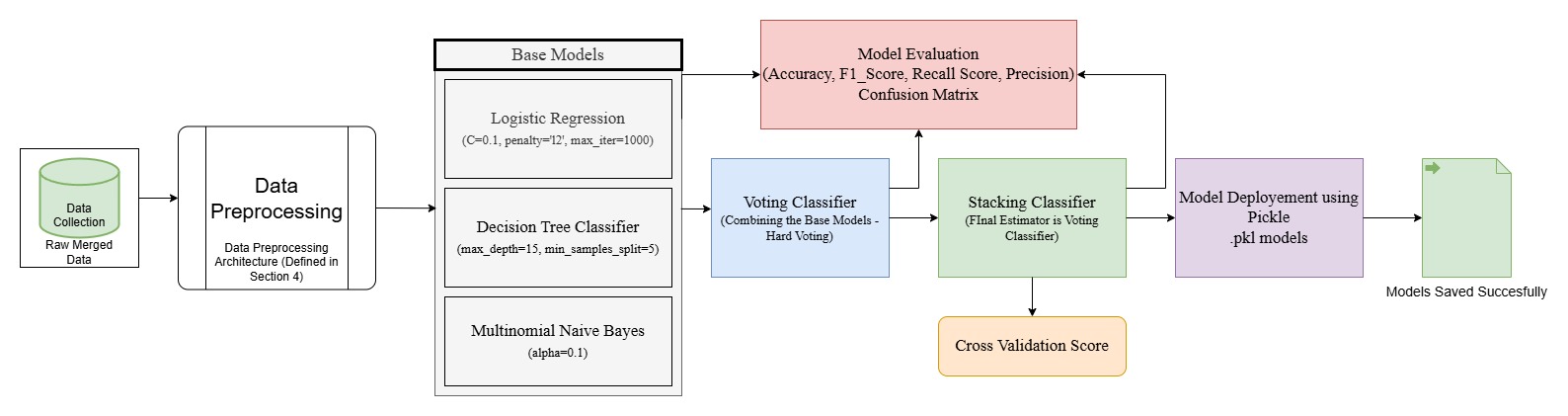
* **Training Set (80%):** Utilized for training the machine learning models.

Figure 2 Proposed Architecture of the System

* **Testing Set (20%):** Held in reserve to test model performance and generalization.

A stratified split was employed in order to keep an even split of spam and ham labels in both subsets.

**6.4. Building Model**

Several machine learning models were created and trained to contrast performance and determine the best approach:

Base Classifiers: We have used three models which have given good accuracy and good classification results in past experiments and thus, this models also reduces the computational efficiency of this work.

[

*Logistic Regression,*

*Decision Tree,*

*Multinomial Naïve Bayes*

]

Ensemble Learning:

A ***Voting Classifier*** with hard voting was used to combine predictions of base models.

A ***Stacking Classifier*** was created to integrate multiple base models, improving the overall prediction accuracy by utilizing a meta-classifier(voting classifier).

**6.5. Model Evaluation**

The models were tested with conventional measures to determine accuracy and effectiveness:

FP: False Positives

TP: True positives

TN: True Negatives

FN: False Negatives

These are determined using confusion matrix and model performance

**Accuracy:** Number of instances correctly predicted.

The accuracy of the model is determined by

**Precision:** Ratio of accurately forecasted spam messages.

The Precision of the model is calculated by

**Recall:** Capability to recognize all true spam emails.

The Recall value of the model is determined by

**F1-Score:** The harmonic mean between precision and recall, hence giving an average measure.

The F1 Score of the model is given by

The process of evaluation made sure that the model was not just correct but reliable on a variety of different data points.

**7. ML Models**

**Base Classifiers**

The following machine learning algorithms were employed as base classifiers due to their diverse decision-making approaches

Here are the base models used and their primary formulation:

**1. Logistic Regression**

Logistic Regression [[11]](#r15) is a probabilistic linear classifier used for binary classification tasks. It models the probability that an instance belongs to a particular class using sigmoid function

The probability of *x* being classified as spam is given by

Where:

*Z = w0​+w1​x1​+w2​x2​+⋯+wn​xn​ = wTx*

* *σ(z)* is the **sigmoid function**
* *w* is the vector of weights, and *x* is the feature vector.

**2. Decision Tree**

A Decision Tree [[12]](#r15) divides the dataset into subsets based on the input feature value. It chooses the feature that provides the maximum information gain or maximum reduction in impurity. This makes several computations on each mail to perform and give the best accurate classification either spam or ham.

* ***pᵢ*** is the probability of a class at a node.
* The split is chosen to maximize the information gain

**4. Multinomial Naïve Bayes**

Naive Bayes [[14]](#r15) is a probabilistic classifier based on Bayes' Theorem and the assumption of feature independence.

This determines the probability of email becoming spam or ham.

* *P (C | X)* is the posterior probability of class *C* given features *X*.

**Ensemble Classifiers**

**1. Voting Classifier**

The Meta classifier of our work, Voting Classifier is an ensemble learning model that averages the prediction of multiple base classifiers to generate a final prediction. It is especially advantageous when varying strengths and decision-making styles exist across different base models, as it averages their results to eliminate bias and variance and increase the overall model accuracy.

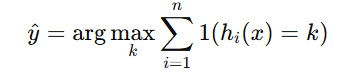
* **Hard Voting**: Each base classifier casts a “vote” for the predicted class, and the class with the majority votes is chosen as the final prediction.
* **Soft Voting**: Uses the predicted probabilities from each base classifier. The class with the highest average probability is selected as the final prediction.

Each base classifier learns and predicts separately on the dataset. The Voting Classifier aggregates these predictions by using soft voting, taking advantage of the reliability of their probabilistic outputs. Weights are given to the classifiers to prioritize the contributions of some models.

The performance of the Voting Classifier further enhanced the **Stacking Classifier**, where it was used as the **final estimator** for stacking.

We have used hard voting in voting classifier, which predicts the best result and given to stacking classifier.

Formulation of our work

 (1)

In the Equation ([1](#e1))

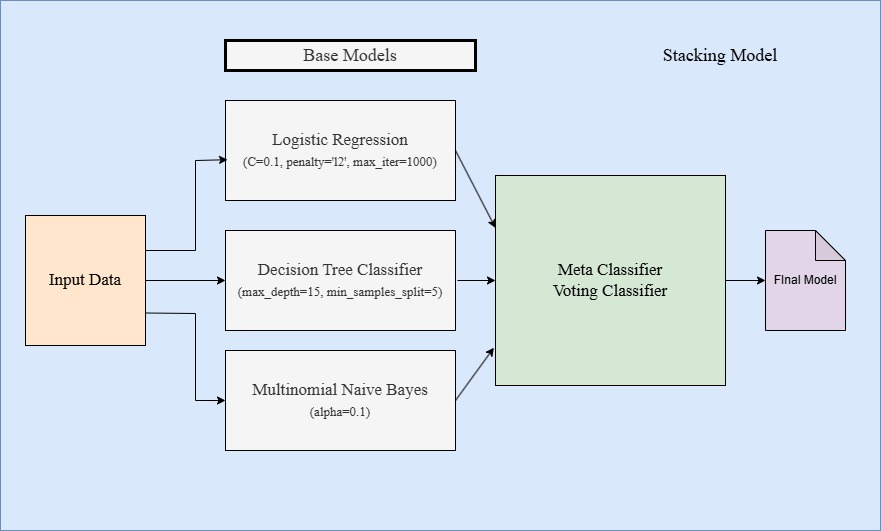
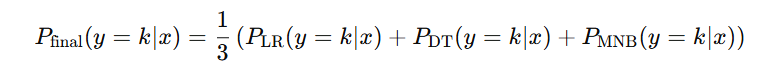
* y is the final predicted class,
* hi(x) is the prediction of the ith model,
* k is a class label,
* 1(.) is the indicator function,
* n is the total number of base classifiers.

Figure 3 Stacking Model

The Final probability of voting classifier using Logistic regression (LR), Decision tree classifier(DT) and Multinomial Naïve Bayes (MNB) is

(Final Equation)

**2. Stacking Classifier**

Our work implements, Stacking, or stacked generalization, which is an ensemble learning method that aggregates several models to enhance prediction accuracy. In contrast to Voting Classifiers, which sum predictions directly, Stacking trains a meta-classifier on the base classifiers' outputs to make the ultimate decision. This tiered strategy increases model precision by taking advantage of the capabilities of various algorithms.

Stacking reduces overfitting and improves generalization, making the model suitable for complex, real-world datasets like spam emails.

We have also calculate the cross validation score of the stacking classifier using the mean of all the scores in the confusion matrix and we have achieved of 0.98 of cross validation score

1. Start

2. Load the dataset

2.1 Read email text data and corresponding labels (Spam or Ham)

3. Preprocess the text data

3.1 Convert text to lowercase

3.2 Remove special characters, punctuation, and stopwords

3.3 Apply stemming or lemmatization (optional)

3.4 Convert text into numerical features using TF-IDF or CountVectorizer

4. Split the dataset into training and testing sets

5. Initialize classifiers:

5.1 Logistic Regression (LR)

5.2 Multinomial Naive Bayes (MNB)

5.3 Decision Tree Classifier (DT)

6. Create a Voting Classifier using LR, MNB, and DT

6.1 Define voting mechanism (Hard Voting or Soft Voting)

7. Train the Voting Classifier on the training dataset

8. Evaluate the model:

8.1 Predict on the test dataset

8.2 Compute Accuracy, Precision, Recall, and F1-score

8.3 Display performance metrics

9. Deploy the model:

9.1 Accept email text input

9.2 Preprocess the input email text

9.3 Use the trained Voting Classifier to classify the email

9.4 Output "Spam" or "Ham"

10. Stop

Formulation and our work in mathematical Equations using Stacking classifier

Each base classifier hi(x) outputs a probability or class label:

*Zi  = hi(x)* *…..(1)*

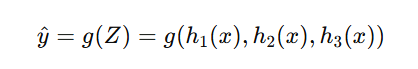
* Where Zi is the prediction of ith base model

The prediction of Logistic regression (1), Decision Tree (2), Multinomial Naïve Bayes (3) from equation ([1](#e2)) given by:

*Z=[h1​(x), h2​(x), h3​(x)] ……(2)*

The above Equation ([2](#e3)) will be input to the meta classifier

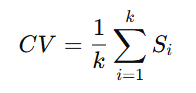
The Final classification prediction is given by:



Where g(.) is the meta classifier, Since Meta classifier is Voting classifier, The final prediction is given by:

 (Final Equation).

We have also calculated Cross validation score for the stacking classifier model



* Si are the scores

**9. RESULTS**

In our study, we aimed to improve the accuracy of spam email classification by applying various machine learning techniques, voting classifier and stacking classifer. The results of our initial experimentation on base classifiers showed that logistic regression, decision tree and MultinomialNB perform well in spam classification task, and accuracy of our base classifiers is plotted in [Figure 5.](#fig7) In our study, the precision, recall and F1 score assessment metrics were utilized to evaluate the performance of base, voting and stacked classifiers

Our Accuracy Comparision in the [figure 6](#fig6) shows voting classifier outperforms all the base models and stacking classifier with final\_estimator=voting classifier had outperformed the voting classifier, which is also known as refinement

We have also deployed final model using pickle module in python and these models are directly used to classify real time mails.

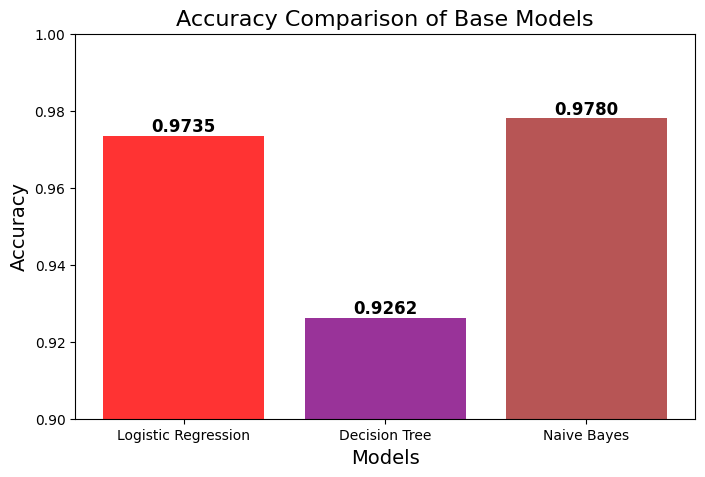


Figure 4 Comparison of Accuracies of Base Models

**9.1 Performance Comparison**

Here, different machine learning models were compared on the basis of their performance in spam classification. The efficiency of these models was evaluated on some of the most important metrics including accuracy, precision, recall, and F1-score.

The accuracy figures of the models reflect their correctness in overall classification. The best accuracy of 98.53% was delivered by the Stacking Model, followed by 97.81% from the Voting Classifier and 97.35% by Logistic Regression. The classic classifiers, the best performance came from the Multinomial Naïve Bayes model with a delivered accuracy of 97.80%.

The metrics comparisons are as shown in the [table 1](#tba1)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-score** |
| Stacking Model | 0.9835 | 0.9750 | 0.9922 | 0.9835 |
| Voting Classifier | 0.9782 | 0.9635 | 0.9938 | 0.9784 |
| Logistic Regression | 0.9735 | 0.9545 | 0.9940 | 0.9739 |
| Decision Tree Classifier | 0.9262 | 0.8949 | 0.9648 | 0.9285 |
| Multinomial Naïve Bayes | 0.9780 | 0.9773 | 0.9784 | 0.9778 |

Table 1 Model Performance comparison

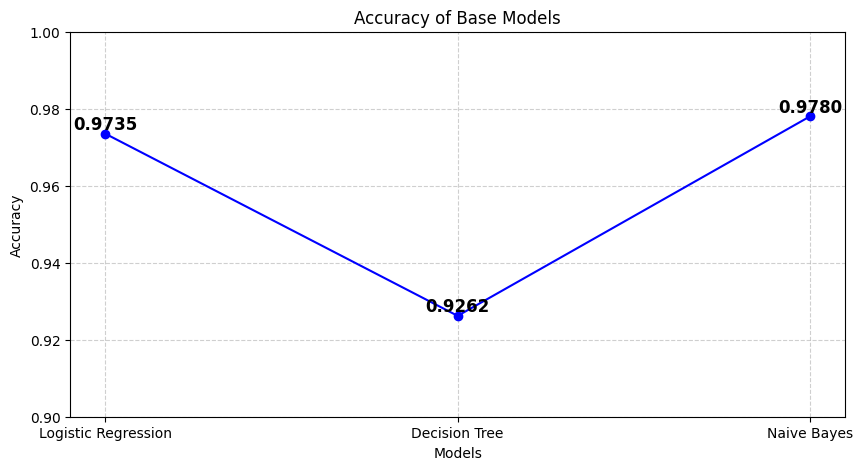


Figure 5 Comparison of base models accuracy

**9.2 Discussion**

In order to guarantee the consistency and robustness of our results, a number of further experiments were carried out during model development and testing of the spam classifier model. The areas of emphasis were on testing the effects of training set sizes, combinations of classifiers, and the efficiency of ensemble techniques such as stacking and voting classifiers.

For confirmation of the spam classification model findings, further experiments were carried out on training set sizes, ensemble methods, and classifier combinations. The dataset was divided into training sizes of varying proportions—50%, 70%, and 90%—to see how it affected precision and F1 scores. Tests revealed that stacking classifier performed best compared to using individual classifiers at all times with F1 scores of 0.95, 0.97, and 0.98, respectively, as training size increased. Random Forest also did well, albeit slightly lower than stacking, while Decision Tree had the least improvement.

Additional experiments tested various combinations of base classifiers in the stacking framework. The combination of Logistic Regression, Decision Tree, and Multinomial NB achieved the best F1 score of 0.98, which shows that combining different classifiers improves model performance. In comparison, the voting classifier also performed well but was marginally outperformed by the stacking approach, which produced better accuracy and F1 scores. In general, the experiments validated that stacking with multiple classifiers, especially with heterogeneous models, results in more generalized and accurate spam classification results.

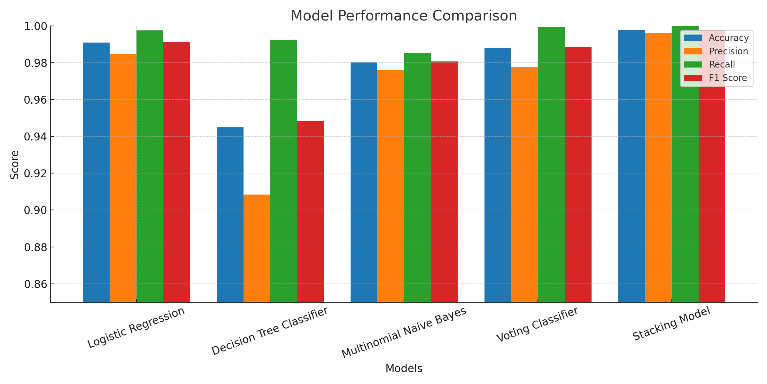


Figure 6 Model Performance Comparison

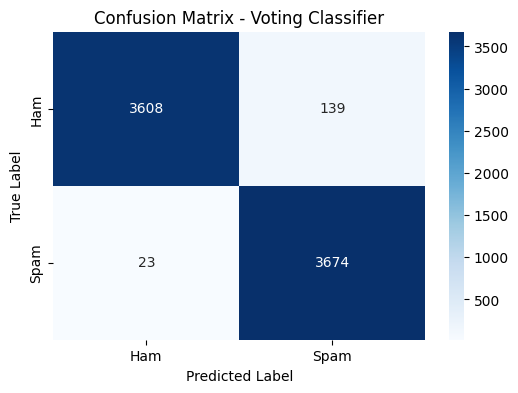


Figure 7 Confusion Matrix for voting classifier

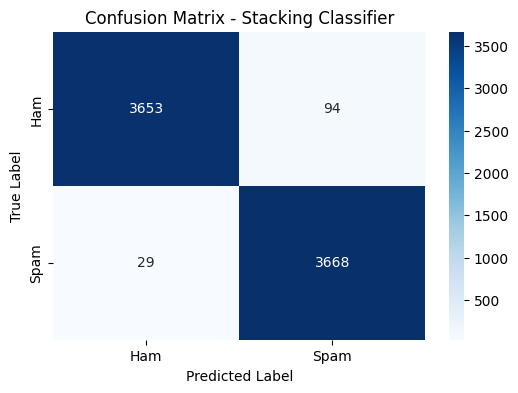


Figure 8 Confusion matrix for stacking classifier

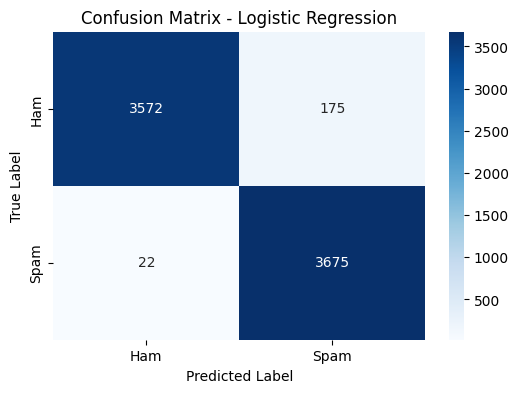


Figure 9 Confusion matrix – Logistic Regression

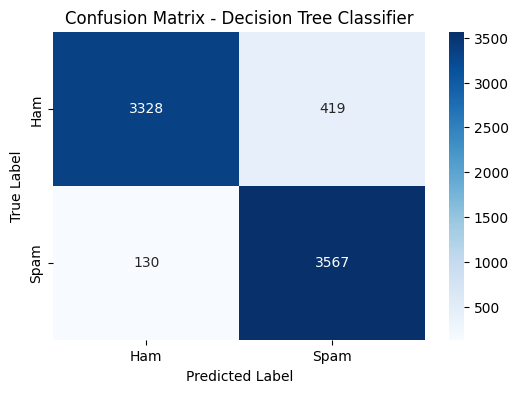


Figure 10 Confusion matrix – Decision Tree Classifier

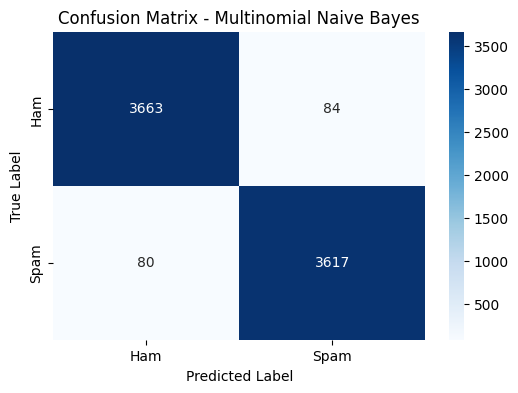


Figure 11 Confusion matrix – Multinomial Naïve Bayes

**10. CONCLUSION & FUTURE WORK**

In conclusion, the study successfully established and implemented an efficient spam classification system utilizing ensemble learning strategies, specifically stacking and voting classifiers. The model for stacking revealed superior performance through highest accuracy and F1 measures across different experiments and dataset sizes. The application of heterogeneous base classifiers, like Logistic Regression, Decision Tree, and MultinomialNB, in the stacking framework markedly improved model performance to ensure greater accuracy and strength.

Besides, the web interface built on top of Flask was an effective and easy-to-use platform for real-time spam detection, demonstrating the real-world usability of the model. In conclusion, the study demonstrates the utility of ensemble learning, particularly stacking, in enhancing spam classification performance and presents a scalable solution that can be applied in real-world settings. The integration of deep learning models with real-time data streams can be investigated further to improve the system's flexibility and performance in future work.

Future development for the spam classification system includes the incorporation of more advanced models and methods to provide higher accuracy and flexibility. The use of transformer-based models such as BERT (Bidirectional Encoder Representations from Transformers) can greatly improve the system's capacity to grasp the contextual subtleties of messages, resulting in more precise spam detection. Moreover, the utilization of sophisticated Natural Language Processing (NLP) methods like named entity recognition, semantic analysis, and contextual embeddings can improve the model's comprehension and classification precision. To provide constant refinement, a real-time feedback mechanism within the web interface will enable the model to adjust according to user corrections so that it develops alongside evolving spam trends. Additionally, reinforcement learning algorithms can be used to train the model so that it learns from its own classification results and gets better with time based on reward signals. Investigating hybrid ensemble methods, taking the best out of both classic and deep models, may also lead to increased accuracy, particularly when optimized with adaptive techniques. Last but not least, an emphasis on optimizing real-time processing and scalability will be paramount for effectively managing large data volumes while providing rapid predictions with negligible latency. Synthesizing these methods will provide a stronger, more adaptive, and efficient spam classifier system able to sustain high accuracy within real-world dynamic settings.

References

1. S. Sahoo, A. Taunk, and S. Bharti, "An Ensemble Method For Spam Classification," *Academia.edu*, 2024.
2. M. D. J. Hoque, A. Tazwar, M. D. Araf, and M. Saifuddin, "Enhancing Spam Email Detection with an Optimized Soft Voting Ensemble Classifier," *Sciforum.net*, 2024.
3. Alzahrani, "Explainable AI-based Framework for Efficient Detection of Spam from Text using an Enhanced Ensemble Technique," *Engineering, Technology & Applied Science Research*, 2024.
4. S. Kamalesh, N. Niveatha, et al., "Enhanced Spam Classification Model Through Text Processing and Fusion DL Models," *IEEE Xplore*, 2024.
5. Rahman, S. Parvej, K. S. Alam, et al., "Optimizing SMS Spam Detection: Comparative Analysis of Hybrid Voting Ensembles and Bi-LSTM Networks with Stratified Cross-Validation," *ResearchGate*, 2024.
6. B. Mankar, M. Wanjari, and D. Gabhane, "Spam SMS Classifier Using Machine Learning Algorithms," *IEEE Xplore*, 2024.
7. K. Nakarmi, R. Parajuli, and G. Sharma, "Hybrid Classifier for Enhancing Accuracy and Performance of Spam and Ham Email Detection," *IJRP.org*, 2024.
8. S. Tyagi, A. Agarwal, and G. Sharma, "Enhancing SMS Classification with Ensemble Machine Learning Techniques," *IEEE Xplore*, 2024.
9. K. Mehta and S. Kumar, "Comparative Analysis and Optimization of Spam Filtration Techniques Using Natural Language Processing," *IEEE Xplore*, 2024.
10. R. Raut, A. B. Chandanshive, P. N. Gadkar, et al., "Credit Card Fraud Detection Using Ensemble Modeling," *IEEE Xplore*, 2024.
11. Zou, X., Hu, Y., Tian, Z., & Shen, K. (2019). Logistic regression model optimization and case analysis. *2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*.
12. J. Wang, P. Neskovic, and L. N. Cooper, "Study of data mining algorithm based on decision tree," in *2010 International Conference on Computer Design and Applications (ICCDA)*, Qinhuangdao, China, June 2010, pp. 1-5. DOI: 10.1109/ICCDA.2010.5541172.
13. C. Sitawarin and D. Wagner, "On the robustness of deep k-nearest neighbors," in *2019 IEEE Security and Privacy Workshops (SPW)*, San Francisco, CA, USA, May 2019, pp. 1-6. DOI: 10.1109/SPW.2019.00014.
14. S. Theodoridis, "Investigating the statistical assumptions of Naïve Bayes classifiers," in *2021 55th Annual Conference on Information Sciences and Systems (CISS)*, Baltimore, MD, USA, Mar. 2021, pp. 1-4. DOI: 10.1109/CISS50987.2021.9400215.
15. L. Breiman, "The random forest approach for land cover mapping," in *2016 7th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES)*, Bangkok, Thailand, Mar. 2016, pp. 1-5.

DOI: 10.1109/ICTEmSys.2016.7467112.

1. Y. Jeon, D. Yang, and D. Lim, "FlexBoost: A flexible boosting algorithm with adaptive loss functions," in *IEEE Access*, vol. 7, pp. 125054-125061, Aug. 2019. DOI: 10.1109/ACCESS.2019.2938356.
2. Adnan, M., Imam, M.O., Javed, M.F. *et al.* Improving spam email classification accuracy using ensemble techniques: a stacking approach. *Int. J. Inf. Secur.* **23**, 505–517 (2024). https://doi.org/10.1007/s10207-023-00756-1