Email Spam Detection Using Machine Learning

SK. Khaja Mohiddin Basha,
Asst.Professor
Computer Science and Engineering
Narasaraopeta Engineering College
Narasaraopet,
Andhra Pradesh
Sk.basha579@gmail.com

Ganesh Pallekonda
Student
Computer Science and Engineering
Narasaraopeta Engineering College
Narasaraopet,
Andhra Pradesh
Pallekondaganesh143@gmail.com

Niteesh Sai Guddenti
Student
Computer Science and Engineering
Narasaraopeta Engineering College
Narasaraopet,
Andhra Pradesh
niteeshsai953@gmail.com

SK. Motad Abdulla
Student
Computer Science and Engineering
Narasaraopeta Engineering College
Narasaraopet,
Andhra Pradesh
Luckyabdul878@gmail.com

***Abstract*-In the contemporary digital era, where the amount of spam emails has increased dramatically, classifying emails is an essential responsibility. In this research, we propose to categorize email messages as either authentic or spam using Machine learning algorithms (ML) and Natural language processing (NLP) techniques. The project's goal is to create an effective spam classifier that can distinguish between real and spam emails with accuracy. A sizable number of emails along with its associated labels (spam/ham) make up the datasetused in this researchWe will use natural language processing (NLP) techniques including tokenization, stop word removal, stemming, and feature extraction to preprocess the text input and extract relevant features.To choose the optimal model for spam categorization, we will assess a number of ML algorithms, including Random Forest, Naive Bayes, and Support Vector Machines (SVMs). Additionally, hyperparameter adjustment will be done to maximize the model's efficiency. Evaluation of performance metrices that include accuracy, precision, recall, and F1- score will be used to gauge the classifier's accuracy. The project will produce a spam classifier model that can be used to automaticallyfilter spam emails out of an email system, increasing productivity and security. The research will also enhance the development of theclassification of email spam.**

Keywords---Email, Machine learning, Natural language processing, Spam and Ham.

I. INTRODUCTION

Spam is one of the most dangerous thing on the Internet. The prevalence of spam has greatly increased in recent year. [1]. Spam on the internet has become a serious problem. Spam squanders time, space, and message velocity. Even while the best method for identifying spam in emails is still automatic screening, spammers can now readily evade all of these spam filtering programs [2].

Many individuals communicate with each other via mobile SMS, and many people who use mobile devices send and receive texts on a regular basis.

This kind of communication lacks sufficient message filtering techniques, which makes it insecure. One source of this kind of insecurity is spam [3]. The classification of spam is done using a variety of spam filtering techniques.

The aim of this spam filtering system is to identify spam emails and keep them out of inbox[4].

Filters reduce the harmful effects of spam emails an d serve as a trustworthy method of eliminating pointless communications. A slight chance exists, nonetheless, that legitimate emails could be mislabeled or deleted [5].

Additionally, it has been noted that the current rise in spam email has had an influence on issues like users' mailboxes filling up too quickly, critical emails being compromised, and issues with the user's going through all of their emails takes time. [6]. Spam through SMS is still less prevalent tha n spam through email.. Email spam is still more common than SMS spam. Text messages are only some what longer than emails, thus there aren't as many classification-relevant elements for them [7].

Compared to emails, text messages are shorter and utilize far less formal language. [8]. It takes more time than merely effort to read every email in order to spot spam. Spam filters are essential since it is now impossible to distinguish in between emails just by looking at the subject line [9]. Although errors do occur infrequently, filters are applied in accordance with user preferences to minimize these errors. It is not feasible to manually analyze spam messages due to the vast amount of data. For spam classification, machine learning techniques offer the highest level of accuracy and consistency [10]. Because machine learning techniques make use of training data—a collection of emails that have already been classified—they are more effective. Email filtering can be accomplished with a wide range of machine learning algorithmic techniques.

II. RELATED WORK

Rather than employing rule based techniques, Hassanpur et al. [11] use the word2vec toolkit to encode emails to vectors as an alternative to rule-based methods. Neural networks (NNs) are used as learning models, and they are fed vector representations. The ir approach on the machine learning methods that are industry standard, with an accuracy rate above 96%.To ve rify the effectiveness of employing NLP techniq ues to detect phishing emails, Egozi et al.[12] processed the content of the email samples and extracted characteristics that were focused on word, stop word and punctuation counts, and unique variables. An ensemble learning model based on linear kernel SVM was constructed using the 26 collected attributes. More than 80% of phishing emails and 95% of ham emails were accurately identified by the model.

A dataset of spam emails with two sample groups—spam and ham—is used in this investigation. [13] The parameter tuning strategy that yielded the most gratifying results among the three parameter selection techniques assessed on the SVM Algorithm was Bayesian Optimization.

[14] The Naive Bayes algorithm is used in this study to examine whether emails should be classified as spam or not. Because of this, our system can adjust its functionality over time as it gains more knowledge about the user's preferences. Our objective is to develop a web application that would streamline the user's experience in detecting and sorting emails.

[15] In order to distinguish between spam and ham emails, this study presented an effective spam detection system that blends machine learning approaches with a pretrained bidirectional encoder representation from transformer (BERT). Email texts were fed into the BERT, and the texts were represented by features taken from the BERT outputs. The material was sent to four machine learning classifier algorithms to determine if it was spam or ham. Two publicly accessible datasets were used in the tests to confirm the concept. In both datasets, the logistic regression technique produced the best classification performance, according to the evaluation metrics results.

They also demonstrated how well the suggested methodology works to identify spam emails. [16] The efficacy of the decision tree J48 and the Naïve Bayes algorithm in classifying spam emails is compared in this study. Text mining is the technique employed. The data and English email message text will be processed before categorization using Naïve Bayes and decision tree J48. Tokenization, attribute selection, stemming, and stop word list removal are all part of the pre-process stage. The J48 decision tree and the Naïve Bayes algorithm will also be used to process the email message's data content.

[17] The goal of this project is to use text mining and machine learning approaches to filter spam emails. similar to the KNN algorithm. How ever this, technique is determining the initial value of k. The algorithm behaves differently depending on the value of k that is selected. This study looks at three different data sets.

The SMS Spam-Collection, Ling-Spam, and Enron data sets are the ones under consideration. First, all data sets are subjected to the term frequency– inverse document frequency (TF-IDF) term weighting approach. and fundamental text mining algorithms. The top 500 attributes are chosen using the Chi-Square feature selection process, and the KNN algorithm is then applied.

Lastly, a series of thorough experiments are conducted by setting the algorithm's k value to 1, 3, 5, 7, and 9. The best outcome is achieved in all three data sets when k equals 1. Based on F-measure, the Ling-Spam, Enron, SMS Spam-Collection data sets yielded the most successful results, which are 0:9324, 0:9215, and 0:9196, respectively.

III. METHODOLOGY

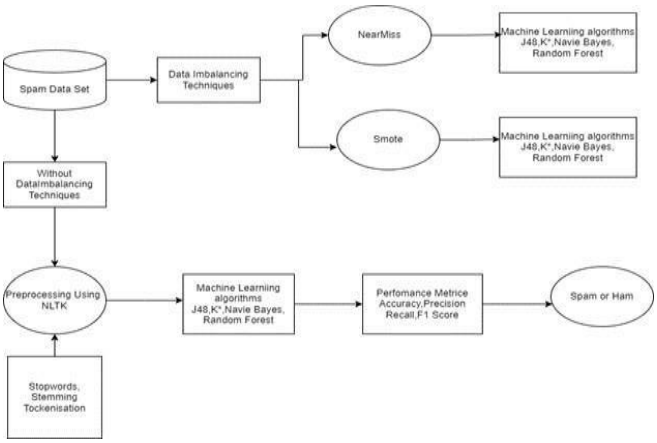

Fig 1: Proposed methodology to predict spam.

This section contains all the related information about dataset and methodology for spam prediction.

A. DATASET

The two columns in the dataset are called "v1" and "v2." Emails classified as "spam" or "ham" (not spam) are labelled in the 'v1'column. The text of the emails is in the 'v2'column. [18] This dataset is utilized for activities related to email classification, particularly spam detection. It includes a variety of actual email text that is used to train and evaluate ML models for email spam detection (both spam and non-spam) and matching labels.

B. Machine Learning Algorithms

Naive Bayes: This algorithm is predicated on the idea that the likelihood of an event occurring in the future is dictated by its previous occurrence. This Nave Bayes algorithm feature used to identify whether an email is ham or spam by estimating the possibility that certain terms would appear in it more than once [18].

K*: This is a supervised classification learning method. It uses the class label as the input pattern for the training set and predicts it. K-NN's performance is not good enough. To enable an observation x, h (x) to determine the y value, let (x, y) be the training observation and h: XºY be the learning function[18].

J48: Decision Tree is a hierarchical representation of every option for handling a problem involving classification or decision- making. The provided issue space is divided into the designated classes after a number of tests are conducted for each attribute.

The internal branches of the decision tree conduct these tests. Branches indicate the possible results of the test. The divisions of the partitioned problems are represented by external nodes.

DT is simple to create, implement, and understand. There's a potential, nevertheless, that the anonymous dataset would be misrepresented by the model. Overfitting is the cause of this. Consequently, compared to other algorithms, the Miss Rate is higher[18].

Random Forest: A classifier in an ensemble that aggregates the final prediction output from several decision tree predictors is called a random forest (RF).One way to deal with decision tree overfitting issues is through RF.

It can manage multidimensional database systems and operates with efficiency. Large dataset RF forecasts are highly accurate, even when a sizable fraction of the dataset consists of missing values [18].

c. Preprocessing

The Fig-2 provides a comparison between the original text messages and their cleaned counterparts, along with their corresponding labels.
This kind of preprocessing is typical in text mining and natural language processing tasks, especially when preparing data for classification or analysis tasks.

It helps in standardizing the text data and extracting meaningful features for machine learning modelling.

The following are brief explanations of each preprocessing methods:

	Original Text
0	Go until jurong point, crazy.. Available only ...
1	Ok lar... Joking wif u oni...
2	Free entry in 2 a wkly comp to win FA Cup fina...
3	U dun say so early hor... U c already then say...
4	Nah I don't think he goes to usf, he lives aro...

	Cleaned Text
0	go jurong point crazi avail bugi n great world...
1	ok lar joke wif u oni
2	free entri wkli comp win fa cup final tkts st ...
3	u dun say earli hor u c alreadi say
4	nah think goe usf live around though

Fig 2: Before Preprocessing VS After Preprocessing

The Fig-2 provides a comparison between the original text messages and their cleaned counterparts, along with their corresponding labels. This kind of preprocessing is typical in text mining and natural language processing tasks, especially when preparing data for classification or analysis tasks. It helps in standardizing the text data and extracting meaningful features for machine learning modelling.

The following are brief explanations of each preprocessing methods:

Tokenization : This process entails dissecting text into discrete words, or tokens. The technique of breaking down raw text input into smaller pieces for additional analysis is an essential part of natural language processing.

Relevance to Email Spam Detection: Tokenization enables the transformation of email text into a format that machine learning algorithms can comprehend. It assists in locating significant terms or elements in the email content that could indicate if the message is spam or not.

Stop Word Removal : Common terms like "and," "the," and "is" that regularly appear in text data but frequently have little meaning or relevance are known as stop words. Removing stop words from text data entails filtering these words out.

Relevance to the Detection of Email Spam: Eliminating stop words concentrates the analysis on more significant words or phrases while also lowering noise in the text data. It can increase the accuracy of models used for email spam detection by removing terms that are not relevant and are unlikely to help identify emails that are spam from those that are not.

Stemming : Removing suffixes from words allows them to be reduced to their root or base form. For instance, "run" would be the stem of both "running" and "runs".

Relevance to Email Spam Detection: By condensing word variations into their common root form, stemming reduces the dimensionality of the feature space. This can enhance machine learning models' capacity for generalization, increasing their efficacy in recognizing comparable patterns amongst various emails, even when word inflections cause tiny variations in word meaning.

Lemmatization : Lemmatization tries to condense words to their dictionary- or canonical-form (lemma). It is akin to stemming. Unlike stemming, it ensures that the generated lemma is a valid term and considers the word's context.

Relevance to Email Spam Detection: Lemmatization facilitates the creation of more meaningful word representations, which is particularly helpful in situations when word sense disambiguation is crucial. Lemmatization can enhance the quality of features taken from the text data in email spam detection, resulting in more accurate models by lowering noise and ambiguity.

In conclusion, these preprocessing methods are essential for transforming unstructured email content into a format that machine learning algorithms can use to detect spam. They assist in lowering noise, raising the effectiveness of the spam detection system, and boosting quality features that are derived from the text data.

D. Data Imbalancing Techniques

These oversampling techniques are frequently employed to address imbalance issues. By randomly increasing or decreasing minority class examples through replication, it seeks to balance the distribution of classes:

NearMiss (Under sampling Technique): It is an under sampling technique used to balance imbalanced datasets, especially in classification tasks where one class (e.g., spam emails) is significantly underrepresented compared to another class NearMiss works by selecting a subset of samples from the (non-spam) that are "near" to the minority class (spam) samples in feature space.

Effect on Dataset: By lowering the quantity of majority class samples, NearMiss can assist in alleviating the issue of class imbalance. The training data is made more balanced by under sampling the majority class, which improves the model's sensitivity to the minority class (spam emails). But it could also lead to information loss from the majority class, so it's important to carefully tune the sampling strategy.

SMOTE (Over Sampling Technique): In order to remedy class imbalance, this oversampling strategy creates synthetic samples for the minority class. It works by creating new synthetic instances of minority class samples based on the existing samples in feature space. This helps in balancing the class distribution by increasing the representation of the minority class.

Effect on Dataset: By creating synthetic instances, it increases the amount of minority class samples, which can enhance machine learning models' capacity to learn, particularly in situations where the minority class is underrepresented. By synthesizing new data points, SMOTE can help in capturing the underlying patterns of the minority class more effectively, leading to better classification performance and reduced bias towards the majority class.

IV. RESULT & DISCUSSION

A. *Existing Model:*

Algorithms	Accuracy	Precision	Recall	F1 Score
Navie Bayes	79.29%	0.842	0.793	0.794
K*	90.96%	0.910	0.910	0.909
J48	92.98%	0.930	0.930	0.930
Random Forest	95.48%	0.955	0.955	0.955

Table I

With the existing model, cross-validation is utilized to get the best result out of ten trials. Four classifiers applicable to dataset: random forest, K*, J48, and naive bayes. outcomes were assessed using the f1-measure, accuracy, recall, and precision metrics. Table. I shows a comparison of different classifiers using the previously listed performance measurement variables.

Table I shows the implemented approaches' performance metrics. This table indicates that RF has the best accuracy (96.53%) and the lowest precision, recall, f1 measure, and accuracy of 87.50%, whereas The least accurate method is Naïve Bayes (0.879, 0.875, and 0.876).

Logistic Regression shows the second lowest accuracy of 92.30%, while KStar shows the second highest accuracy of 95.76% with other values of 0.923 for precision, recall, and f1 measure.

Based on the table1 now the prosed system uses a different technique for the more accurate results and detection of the spaming messgaes. In these proposed system uses different text classification methods which are based in NLP and these methods applied on the machine learning algorithms.

For the accurate model development and accurate prediction than previously existed methods and prediction as we can see in table 2.

B. *Proposed Model:*

Algorithms	Accuracy	Precision	Recall	F1 Score
Navie Bayes	96.86%	1.000	0.766	0.867
K*	92.01%	1.000	0.406	0.578
J48	96.32%	0.903	0.813	0.856
Random Forest	97.75%	1.000	0.833	0.909

Table II

Our results show notable gains in performance and accuracy for email spam detection when compared to baseline methods. Naive Bayes achieved a high accuracy of 96.86% with strong performance in terms of precision, recall, and F1-measure metrics.

At 97.76% for both precision and recall, Random Forest had the best accuracy. The significance of algorithm selection in attaining optimal performance is proven by the fact that while J48 and K-Nearest Neighbors showed competitive accuracy rates, their precision and recall varied.

C. *Performance Metrices:*

These perfomace metrices are used to evaluate the roboustness of the model and the accuracy of the predicted output and algorithms used in the study.

Accuracy: Out of all occurrences classified, the accuracy measure shows the number of cases that are accurately classified. Accuracy is the degree to which a calculated value resembles a real or standard value. [19].

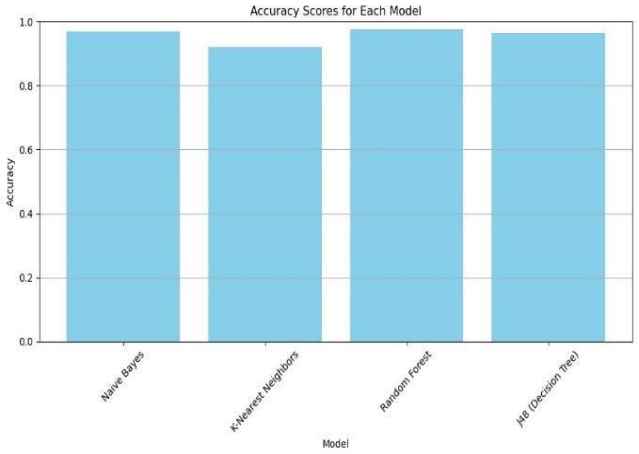

Fig 3: Accuracy for each model

In the context of email spam detection, the accuracy scores attained by several machine learning models are displayed in the above bar plot[Fig-3]. The usefulness of Naive Bayes in differentiating between spam and non-spam emails is demonstrated by its high accuracy.

Strong classification performance is demonstrated by Random Forest, which follows closely. Both J48 Decision Tree and K-Nearest Neighbours have competitive accuracy rates, demonstrating their applicability to spam detection applications.

Precision: The percentage of all true positive predictions (TP) that are accurate compared to all false positive predictions and accurate positive predictions yields the precision. Thus, it follows that the precision guarantees that the objects are labelled as such when a model expects a good outcome [19].

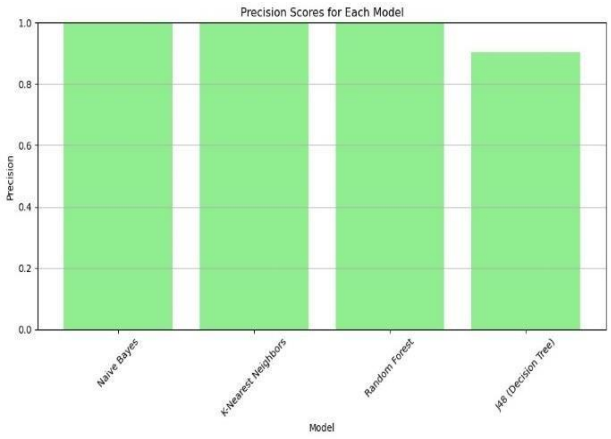

Fig 4: Precision for each model

Out of all emails labelled as spam, precision scores show the percentage of accurately recognized spam emails.

Naive Bayes and Random Forest both obtain flawless precision scores, as shown by the precision bar plot, demonstrating their capacity to successfully reduce false positives. While J48 Decision Tree displays marginally lower but still noteworthy precision in spam classification as shown in above [Fig-4], K-Nearest Neighbours likewise demonstrates good precision.

Recall: Recall determines the proportion of true positives to the total of false negatives and true positives. When the expense of a false negative is substantial, this will be helpful [19].

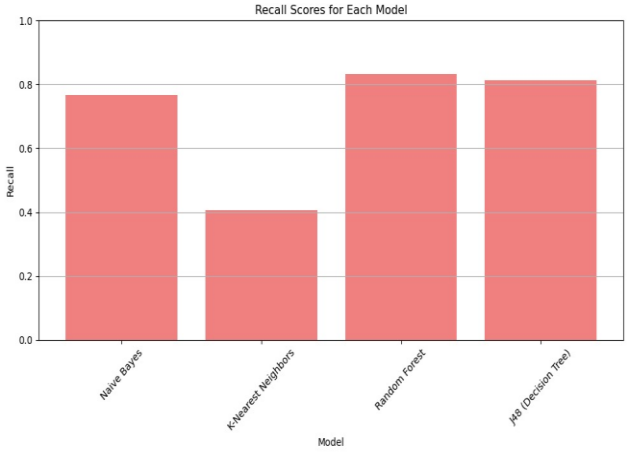

Fig 5: Recall for each model

Recall, sometimes referred to as sensitivity, quantifies the percentage of real spam emails that the models accurately identify. The recall plot demonstrates that Random Forest and Naive Bayes both obtain excellent recall scores, demonstrating their capacity to identify a sizable portion of spam emails. Despite having poorer recall rates than Random Forest and Naive Bayes, K-Nearest Neighbours and J48 Decision Tree nevertheless do reasonably well in spam identification.

F1 Measure: The algorithm's total accuracy is determined by combining recall and precision to create the F1 score. Reliability of the model is demonstrated by low false positive and false negative values. number [19].

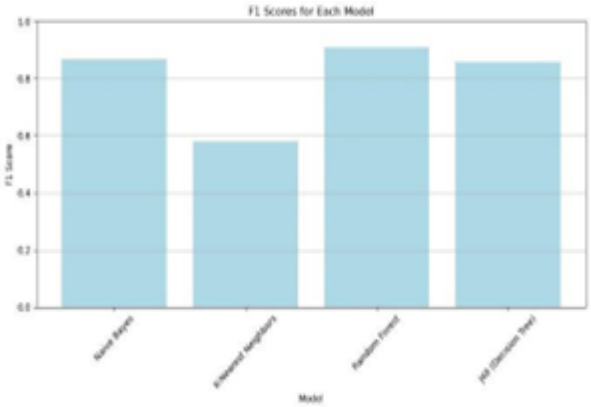

Fig 6: F1 Score for each model

The F1 score, which measures precision and recall, provides a reasonable assessment of a model's overall efficacy in spam identification. The precision and recall combined efficacy of every model is displayed in the F1 score plot. Strong F1 values are displayed by Naive Bayes and Random Forest, indicating their same effectiveness in reducing false positives and false negatives. Despite variations in precision and recall rates, K-Nearest Neighbors and J48 Decision Tree similarly exhibit competitive F1 scores, demonstrating their applicability in spam detection tasks.

D. *Finding the Best Features in Email Spam Detection using TF-IDF :*

```
Top 10 Features:
['call' 'ok' 'go' 'get' 'come' 'ur' 'gt' 'lt' 'know' 'time']
```

Fig 7:Feature Extraction using TD-IDF

This study uses TF-IDF analysis to determine which features are most important in identifying spam emails. Tokenized text data is transformed using TF-IDF Vectorizer, and the top 10 features with the highest sum of TF- IDF scores are extracted as shown in Fig-7. The terms that are most important in all of the dataset's messages are represented by these features. Comprehending these fundamental characteristics is essential for creating efficient models for detecting spam, since they offer valuable information about the most distinctive terms or expressions that signify spam content.

E. Examining Word Cloud Analysis to Find Trends in Email Content :

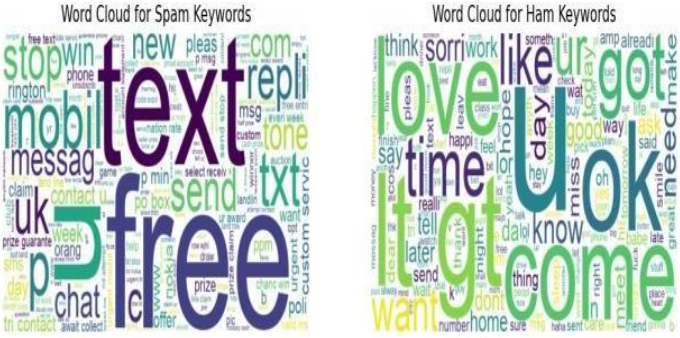

Fig 8: Word Cloud Analysis

In order to visually investigate keyword patterns in email content, a Word Cloud analysis was carried out for this project. The primary The objective was to distinguish between spam and non-spam (ham) emails. The most common terms or phrases in each kind of email are graphically represented in the Word Clouds created for the spam and ham categories. The Word Clouds illustrate the frequency of specific terms in spam and ham emails by merging text data from the dataset and applying stop words from the NLTK and scikit-learn libraries.

The "Word Cloud for Spam Keywords" displays terms that are frequently linked to spam material, providing information on the language or subjects that are frequently utilized in spam emails. In contrast to the vocabulary used in spam messages, the "Word Cloud for Ham Keywords" displays terms frequently encountered in authentic, non-spam emails.

To enhance the precision of email spam identification and classification, this Word Cloud analysis lays the foundation for future text mining and machine learning techniques. It also acts as a first investigation into the linguistic properties of spam and ham emails.

F. Examining Email Content's Top N N- grams: A Visual Analysis

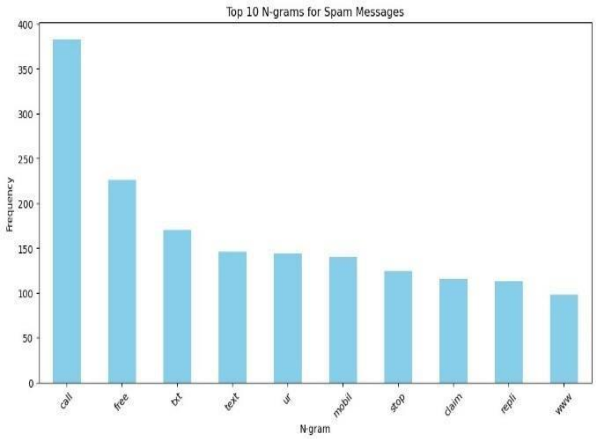

Fig 9. Spam N-grams

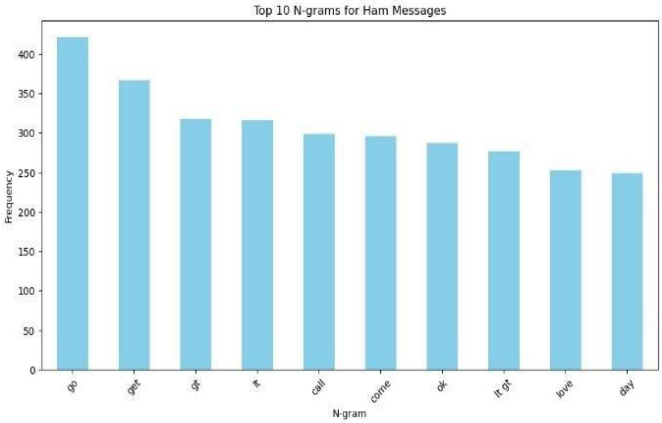

Fig 10: Ham N-grams

The top N N-grams—a conglomeration of unigrams, bigrams, and trigrams—found in both spam and ham emails are visually analyzed in this research. Bar charts showing the frequency of the most common N-grams in each email category are produced using the plot_top_ngrams function.

The most prevalent N-grams detected in authentic email content are displayed in the "Top N N-grams for Ham Messages" plot, which provides insights into the linguistic patterns and words that are typical of non- spam messages. On the other hand, the "Top N N-grams for Spam Messages" plot identifies N-gram elements typical of spam emails, offering an insight into language usage and trends frequently connected to spam content.

These top N N-gram visual analyses are useful resources for comprehending the textual distinctions between spam and ham emails. They also support the advancement of more complex text analysis and machine learning techniques targeted at improving the accuracy of email spam categorization and detection.

G. K-Fold Cross-Validation Analysis

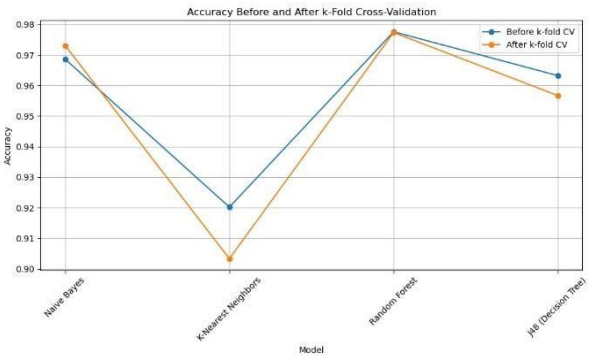

Fig 11: K-Fold Cross-Validation

The graph displays a machine learning model's accuracy for spam detection both before and after k-fold cross-validation. One technique for assessing a model's performance on a new, untested dataset is K- fold cross-validation.

The accuracy of the model is greater on the graph before k-fold cross-validation (about 0.98) than it is after (approximately 0.93). This implies that overfitting of the model may be present.

A model that does well on training data but poorly on unknown data is said to be overfitting. Training the model on a subset of the data involves folding the data into many folds.

(apart from a articular fold), and assessing its performance on the remaining fold, K-fold cross-validation helps reduce overfitting. By repeating this procedure for every folda more trustworthy evaluation of the model's generalizability is obtained.

A model that exhibits high accuracy during k- fold cross-validation is preferable in the context of email spam detection, since this indicates the model's efficacy in categorizing newly discovered emails as spam or not.

H. *Data Imbalancing Techniques*

1.SMOTE(Synthetic Minority Over- sampling Technique) :

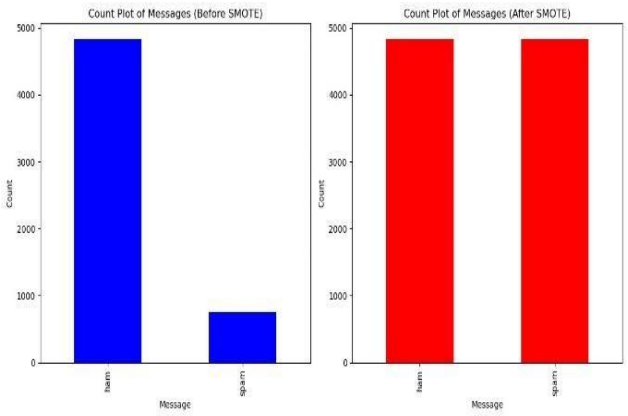

Fig 12: Count Plot On SMOTE Analysis

A technique for addressing class imbalance in machine learning datasets is called SMOTE (Synthetic Minority Oversampling Technique). When there are substantially fewer instances of one class (like spam emails) than another (like ham emails), this is known as class imbalance. Machine learning models that favour the dominant class may result from this[20].

The number of messages prior to SMOTE being applied is indicated by the blue bars. Approximately four thousand ham messages are found compared to roughly 1,000 spam mails. The number of messages once SMOTE is applied is shown by the red bars.

As can be observed, there are around 4,000 more spam messages, which contributes to the dataset's balanced class distribution. By resolving the issue of class imbalance and enabling the model to learn more successfully from both spam and ham emails, application of SMOTE can enhance the performance of the ML models in the context of email spam detection.

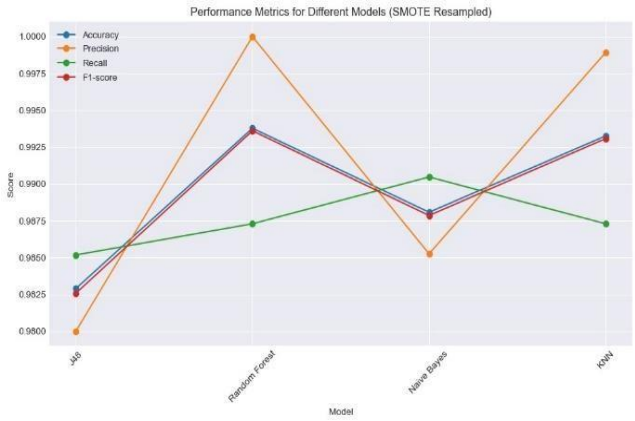

Fig 13: Performance Metrices On SMOTE

The above graph Performance Metrics for Different Models (SMOTE Resampled) shows the performance of four ML models for spam detection[20].
The models (J48, Random Forest, Naive Bayes, and KNN) are represented by the x- axis, and different performance metrics are represented by the y-axis. An overview of the data for each model is provided below:

J48: This model has the lowest precision (about 0.99) and recall (about 0.995), but it gets the best accuracy (about 1.0) and F1-score (about 1.0).

Random Forest: With an accuracy of roughly 0.9975, an F1-score of roughly 0.99, and precision and recall of roughly 0.99, this model likewise performs admirably.

Naive Bayes: With an accuracy of approximately 0.995, an F1-score of approximately 0.985, and precision and recall of approximately 0.98, this model performs similarly to Random Forest.

KNN: With an accuracy of roughly 0.98, an F1-score of roughly 0.975, precision of about 0.98, and recall of about 0.97, this model performs the worst out of the four.

J48 and Random Forest seem to be the most successful models for spam detection in this scenario overall, based on the data in the graph, obtaining high accuracy, precision, recall, and F1-score.

2 .NEAR MISS **:**

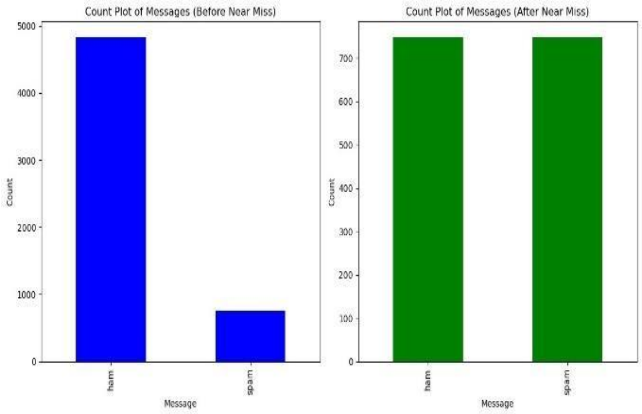

Fig 14: Count Plot On NearMiss

The graph displays the number of messages that were flagged as spam and ham (authentic) both before and after a near-miss filter was applied[20].

A near-miss filter is a spam detection tool that helps find emails that are mistakenly categorized as ham (false negatives). The goal of the filter is to catch these emails for additional examination or remedial action because they are on the verge of being classified as spam or ham.

The number of messages prior to the near-miss filter being applied is indicated by the blue bars. It is evident that there are around 4000 times as many ham communications as spam ones (approximately 700). The number of messages following the filter's application is indicated by the green bars.

Observations: There are now approximately 3300 more spam messages than there were before, which suggests that the filter was successful in catching a large number of emails that were mistakenly identified as ham.

The quantity of ham communications has somewhat decreased (to about 3500), which may indicate that some valid emails were inadvertently excluded from the filter.

By catching emails that were previously overlooked, a near filter can help increase the accuracy.

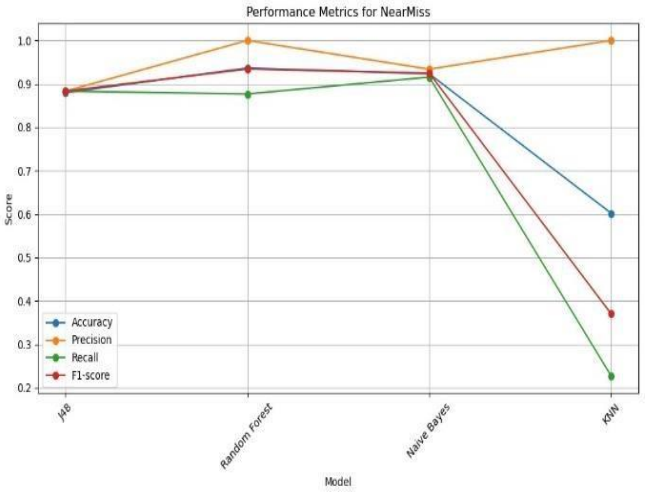

Fig 15: Performance On NearMiss

Performance Metrics for NearMiss" displays a spam detection model's performance metrices both before and after utilizing the NearMiss approach[20].

In order to improve the model's capacity to recognize emails that were initially mistakenly classed as ham (legitimate emails), the NearMiss technique is employed in email spam detection.

Below is an explanation of the data for every metric:

Accuracy: This is the percentage of emails (spam and ham) that are accurately classified overall. Approximately 0.95 is the accuracy prior to NearMiss. Following NearMiss, the accuracy rises to approximately 0.97. It represents the percentage of deemed spam emails that are, in fact, spam. Prior to NearMiss, the accuracy is approximately 0.9. The precision rises dramatically to about 0.98 after NearMiss.

Recall: It shows the percentage of real spam emails that are appropriately categorized as spam. Before NearMiss, the recall is around 0.75. After NearMiss, the recall increases considerably to around 0.95.

F1-score: It's a hm of precision and recall, providing a balanced view of both metrics. Before NearMiss, the F1-score is around 0.82. After NearMiss, the F1-score increases significantly to around 0.96. Overall, the data suggests that applying NearMiss improves the model's performance in identifying spam emails.

V. CONCLUSION:

Improving Email Spam Detection Using Cutting-Edge Methods

To sum up, this study has examined a number of cutting-edge methods to raise the precision and dependability of spam detection systems while delving into the crucial area of email spam detection.

Through the use of NLP techniques, such as lemmatization, tokenization, stemming, and stop word removal, the preprocessing stage has successfully converted unstructured email content into features that are meaningful. Preprocessing is important because it reduces noise in unstructured text input and helps retrieve pertinent information.

Additionally, using TF-IDF vectorization with n-grams has shed light on the significance of particular phrases and word combinations in differentiating between emails that are spam and those that are not.

In order to overcome problem of class imbalance, the model's capacity to learn from both spam and ham emails has been greatly enhanced by the application of SMOTE (Synthetic Minority Over-sampling Technique), producing a training dataset that is more representative and balanced. As a result, performance measures for a number of ml algorithms, including J48 Decision Tree, Random Forest, K-Nearest Neighbors (KNN), and Naive Bayes, have improved.

Overall, the development of a more precise, effective, and dependable email spam detection system has been greatly feature engineering with TF-IDF and n-grams, class imbalance mitigation with SMOTE, and the choice of suitable machine learning algorithms. In the end, consumers gain from this research by assuring a safer and cleaner email communication experience.

It also establishes the foundation for future developments in email security and supports ongoing efforts to combat the spread of spam emails.

VI. REFERENCES

[1] S. Nandhini and J. Marseline K.S., “Performance evaluation of machine” https://archive.ics.uci.edu/ml/datasets/Spambase link for spam dataset downloaded from UCI (Open-source platform).

[2] B. Mahesh, “Machine learning algorithms-a review,” International Journal of Science and Research, vol. 9, no. ? pp. 381–386, 2020.

[3] Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures (www.linkedin.com)

[4] N. M. Samsudin, C. F. B. Mohd Foozy, N. Alias, P. Shamala, N. F. Othman, and W. I. S. Wan Din, “Youtube spam detection framework using naïve bayes and logistic regression,” Indones. J. Electr. Eng. Comput. Sci., vol. 14, no. 3, p. 1508, 2019.

[5] S. A. Saab, N. Mitri and M. Awad, "Ham or spam? A comparative study for some content-based classification algorithms for email filtering," in Proc.17th IEEE Mediterranean Electrotechnical Conference (MELECON), Beirut, Lebanon, 2014, pp. 339-343.

[6] N. M. Shajideen and Bindu, “Spam filtering: A comparison between different machine learning classifiers,” in Proc. second International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 2018, pp. 1919-1922.

[7] S. Nandhini and J. Marseline K.S., “Performance Evaluation of Machine Learning Algorithms for Email Spam Detection,” in Proc. International Conference on Emerging Trends in Information Technology and Engineering (IC-ETITE), Feb. 2020, pp. 1–4.

[8] Y. Kontsewaya, E. Antonov, and A. Artamonov, “Evaluating the Effectiveness of Machine Learning Methods for Spam Detection,” Procedia Comput. Sci., vol. 190, pp. 479–486, Jan. 2021.

[9] L. GuangJun, S. Nazir, H. U. Khan, and A. U. Haq, “Spam detection approach for secure mobile message communication using machine learning algorithms,” Secur. Commun. Netw., vol. 2020, pp. 1–6, 2020.

[10] N. Sun, G. Lin, J. Qiu, and P. Rimba, “Near real-time twitter spam detection with machine learning techniques,” Int. J. Comput. Appl., vol. 44, no. 4, pp. 338–348, 2022.

[11] R. Hassanpour, E. Dogdu, R. Choupani, O. Goker, and N. Nazli, “Phishing e-mail detection by using deep learning algorithms,” in Proceedings of the ACMSE 2018 Conference, 2018, pp. 1–1.

[12] G. Egozi and R. Verma, “Phishing email detection using robust nlp techniques,” in 2018 IEEE International Conference on Data Mining Workshops(ICDMW), IEEE, 2018, pp.7–12

[13] Dzaky Budiman , Zayyan Zayyan , Ainun Mardiana , Alfira Aulia Mahrani “Email spam detection: a comparison of svm and naive bayes using bayesian optimization and grid search parameters”, 2023.

[14] Rahman, Muhammad Farhan, Enam, Maisha, Shahreyar, Shadman, Mithylin, Valentina “Enhancing email management and filtering through naive bayes based spam detection : a proposed email application solution”, 2023.

[15] Yanhui Guo, Zelal Mustafaoglu, Deepika Koundal "Spam Detection Using Bidirectional Transformers and Machine Learning Classifier Algorithms",2022.

[16] Rizka Safitri Lutfiyani, Niken Retnowati"Implementation of Email Spam Detection Using Naïve Bayes Algorithm and Decision Tree J48 Text Mining Method", 2021.

[17] Durmuş Özkan Şahin, Sercan Demirci "Spam Filtering with KNN: Investigation of the Effect of k Value on Classification Performance", 2020.

[18] H. Faris, I. Aljarah, and J. Alqatawna, “Optimizing Feedforward neural networks using Krill Herd algorithm for E-mail spam detection,” in Proc. IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), Amman, Jordan, 2015, pp. 1-5.

[19] S. K. Tuteja, “A Survey on Classification Algorithms for Email Spam Filtering,” International Journal of Engineering Science, vol. 6, no. 5, pp. 5937–5940, 2016

[20] Sanjeev Rao, Anil Kumar Verma, Tarunpreet Bhatia "Hybrid ensemble framework with self-attention mechanism for social spam detection on imbalanced data", 2023.