Volume 11, Issue 1, January 2024 ISSN: 2348-8565 (Online)
International Journal of Modern Engineering and Research Technology
Website: http://www.ijmert.org Email: editor.ijmert@gmail.com

Enhancing Email Security: Cross-Validated Machine Learning for Spam Prediction

Ajeet Kumar Kachhi
Research Scholar, M.Tech.
Computer Science and Engineering
Takshshila Institute of Engineering and Technology
Jabalpur (M.P), India
Email: ajeetkachhee@gmail.com

Swati Soni
Assistant Professor
Department of Computer Science and Engineering
Takshshila Institute of Engineering and Technology
Jabalpur (M.P), India
Email: swatisoni@takshshila.org

ABSTRACT

Even in the era of digital communication, email is still a vital tool for sharing information. Still, the enormous number of spam emails has made the creation of reliable spam Prediction systems necessary. The binary classification of emails into two categories— "Spam" and "Ham" (non-spam) —is the main focus of this study. Our goal is to apply different machine learning models to correctly classify incoming emails and improve the user's email experience using a dataset of 5572 email samples.

Two columns make up the dataset, which was obtained from Kaggle: "Message" and "Category," which have the values "Ham" and "Spam." Our methodology consists of two main steps that investigate various machine learning models and feature extraction techniques. Using a Train-Test split, we use 80% of the data for training and 20% for testing in the first stage. We use the "CountVectorizer" algorithm for feature extraction, which counts the terms that appear in each page. As a result, a matrix of raw term frequencies is produced, which the classification models use as input.

Next, the training and testing accuracies of a group of classification models—Logistic Regression, Decision Trees, Random Forest, Support Vector Machines (SVM), K-Nearest Neighbours (KNN), Gaussian Naive Bayes (GNB), and XGBoost—are assessed. The effectiveness of these models for the binary classification problem is highlighted in this examination. Findings show that SVM performs well on the test set and is a serious contender for spam email identification.

The "TfidfVectorizer," which takes into account both term frequency within a document and the inverse document frequency across the dataset, is used in the second step of our process to extract features. This strategy is made possible by preprocessing the dataset and applying label encoding to the "Category" column. For model evaluation, a cross-validation strategy is used, which offers a more thorough comprehension of model performance. The average accuracy ratings of the several models are calculated, and the results show that Random Forest performs exceptionally well in cross-validation, suggesting that it has potential for the identification of spam emails.

Keywords:—*Email Classification, SpamPrediction, Machine Learning Models, Cross-Validation, Text Feature Extraction, Data Analysis.*

I. INTRODUCTION

Spam Mail- Email communication is still a vital component of contemporary

commercial and interpersonal relationships in the digital age. But this ease is not without its drawbacks, the most enduring of which is the constant barrage of uninvited, pointless, or even dangerous emails—a.k.a. "spam." Spam emails can clog inboxes, waste time, and present security problems because they frequently contain ads, fraudulent schemes, or other undesirable content.

The fields of email classification and spam Prediction have attracted a lot of interest in an effort to lessen the negative effects of spam emails and improve user experience. Email service providers, businesses, and consumers all want efficient ways to automatically distinguish and separate spam emails from real emails, or "ham" emails.

In this endeavour, machine learning has shown to be a highly effective technique. Systems that can differentiate between spam and ham emails can be created by utilising machine learning models, giving consumers access to a more secure and hygienic email experience. By using labelled datasets that include examples of both spam and ham emails, these models are trained to identify patterns and traits that distinguish them apart from the other category. The goal of this project is to perform binary classification, which divides emails into two groups: "Spam" and "Ham." The main goal is to evaluate how well different machine learning models perform in this classification problem, with a focus on cross-validation to guarantee reliable findings. The research makes use of a real-world dataset with 5572 emails that have been classified as either spam or ham. The study's machine learning models use a variety of methods, such as XGBoost, K-Nearest Neighbours (KNN), Gaussian Naive Bayes (GNB), Random Forest, Decision Trees, Support Vector Machines (SVM), and Logistic Regression. This study attempts to shed light on which models are best at detecting spam by analysing these models' training and testing accuracies and then calculating their mean accuracy using cross-validation. The results of this study are important for email service providers who want to improve their spam filtering systems as well as for people and businesses who want to lessen the amount of spam that reaches their inboxes. This project adds to the ongoing work to enhance email security and the user experience by utilizing machine learning and cross-validation.

Machine Learning - A branch of artificial intelligence (AI) known as machine learning gives computers the ability to learn from data and make predictions or judgments without the need for explicit programming. It covers the following three main categories of learning:

1. ***Supervised Learning:*** Using labelled datasets, algorithms are trained to associate input data with desired results. Establishing a mapping between inputs and outputs will allow the model to accurately anticipate new, unobserved data.

2. ***Unsupervised Learning:*** Unsupervised learning looks for patterns, structures, or correlations in datasets that have not been labeled. Reducing data dimensionality and grouping comparable data points are frequent activities.

3. ***Reinforcement Learning:*** The goal of reinforcement learning is to teach algorithms to make choices in a given environment by maximizing cumulative rewards over an extended period of time. The model is taught by the environment, which gives it feedback in the form of incentives or

punishments depending on what it does.

Machine learning encompasses two primary types of tasks:

1. ***Classification:*** Assigning input data points to distinct groups or classes is a type of supervised learning. The model is appropriate for tasks involving categorical output variables because it learns to map input data to specified output classes.

2. ***Regression:*** Predicting continuous numerical values from input data is the goal of regression, another kind of supervised learning. Regression tasks are beneficial for tasks involving continuous output variables because the model learns to build a link between the input characteristics and the output variable.

Given the nature of the Category Column in Dataset "mail_data.csv", the appropriate approach to solve this problem is Classification.In proposed work following Machine Learning Classification Algorithms are used–

- Logistic Regression
- Decision Tree Classifier
- Random Forest Classifier
- Support Vector Classifier
- K Neighbors Classifier
- GNB (Gaussian Naive Bayes Classifier)
- XGB (Extreme Gradient Boosting)

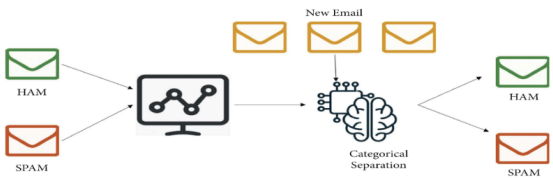

Figure 1. Spam Mail Prediction using Supervised Learning

Feature Extraction - The term "feature extraction" describes the procedure that converts the unprocessed text data from the email messages into numerical features suitable for use as machine learning model input. In order to transform unstructured text input into a format that machine learning algorithms can handle efficiently, this step is essential.

Another method in natural language processing (NLP) for converting a set of text documents into a numerical representation suitable for machine learning applications is called CountVectorizer. CountVectorizer just counts the frequency of each term in each document, in contrast to TfidfVectorizer, which uses TF-IDF scores to account for the relevance of terms within a document and throughout a corpus."Term Frequency-Inverse Document Frequency Vectorizer," or "TfidfVectorizer," is a widely used method in text mining and natural language processing. It is used to transform a set of unprocessed documents into a numerical format suitable for use by machine learning algorithms.

1. ***Term Frequency (TF):*** This method gauges the frequency with which a phrase or word appears in a document. The computation involves tallying the occurrences of a term in a document and subsequently dividing the result by the total number of terms included in the text. This makes it easier to determine a word's significance inside a given document.

2. ***Inverse Document Frequency (IDF):*** This section quantifies a term's importance throughout a corpus, or group of documents. The computation involves dividing the total number of documents in the corpus by the number of documents that contain the phrase, and then computing the

logarithm of that number. Terms that are distinctive to a document will have a higher IDF value than terms that are common in multiple papers, which will have a lower IDF value.

3. ***TF-IDF:*** A term's TF-IDF score is determined by multiplying its IDF by its TF in a document. This yields a score that accounts for the term's relevance inside the corpus as well as its frequency within the document. Higher TF-IDF scores indicate that a term is more significant for that particular document.

Text documents are transformed into a matrix of TF-IDF features via the TF-IDF vectorizer, where each row denotes a document and each column a distinct term throughout the corpus. Then, by using these numerical features as input for machine learning models, text-based tasks like information retrieval, sentiment analysis, and document categorization can be completed.

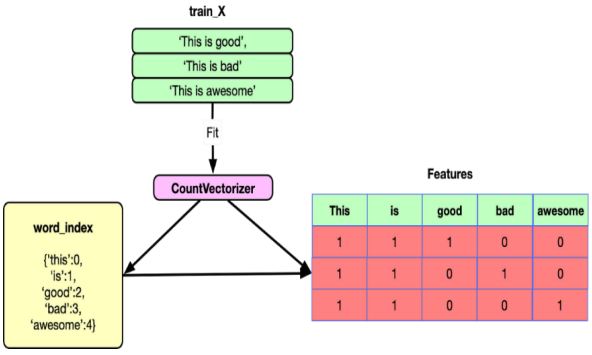

Figure 2. Count Vectorizer

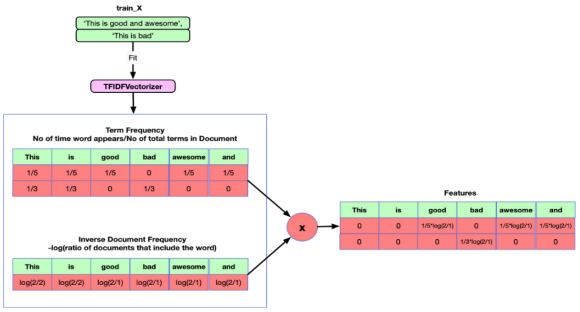

Figure 3.TF-IDF Vectorizer

TF-IDF is better than Count Vectorizers because it not only focuses on the frequency of words present in the corpus but also provides the importance of the words. We can then remove the words that are less important for analysis, hence making the model building less complex by reducing the input dimensions.

Train Test Splitting - In machine learning, the train-test split is a crucial concept used to evaluate the performance of a model and to prevent overfitting. Here's how it works:

1. ***Data Splitting:*** The available dataset is divided into two subsets: - Training Set: This portion of the data is used to train the machine learning model. The model learns patterns and relationships from this data.

   - ***Test Set:*** This portion of the data is kept separate from the training process. It is used to evaluate the model's performance and assess how well it generalizes to new, unseen data.

2. ***Purpose***: Training Set: Used to train the model's parameters (weights and biases in case of neural networks, coefficients in case of linear models, etc.).

   - ***Test Set:*** Used to evaluate the model's performance on unseen data. This helps assess how well the model will perform in real-world scenarios.

3. ***Preventing Overfitting:*** Overfitting occurs when a model learns to memorize the training data instead of learning general patterns. By evaluating the model on a separate test set, we can determine if it has overfit to the training data.

4. ***Split Ratio:*** The ratio between the size of the training set and the test set

can vary depending on the size of the dataset and the specific problem. Common splits include 70-30, 80-20, or 90-10, where the first percentage represents the portion of data allocated to training and the second percentage represents the portion allocated to testing.

5. ***Randomization:*** It's crucial to randomly shuffle the dataset before splitting to ensure that the data in both sets is representative of the overall distribution. This helps prevent any bias in the data splitting process.

6. ***Cross-Validation:*** Sometimes, a single train-test split may not be enough to provide a reliable estimate of a model's performance. In such cases, techniques like k-fold cross-validation are used. In k-fold cross-validation, the data is divided into k subsets, and the model is trained and tested k times, with each subset used as the test set once and the rest as the training set.

By using the train-test split technique, machine learning practitioners can develop models that generalize well to unseen data, thus making them more reliable for real-world applications.

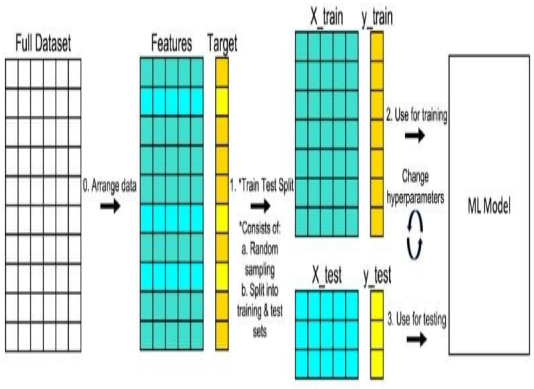

Figure 4. Train Test Splitting

K- fold Cross Validation - The traditional approach of assessing machine learning models, known as train-test splitting, has built-in drawbacks. The unpredictable nature of performance evaluation resulting from the random selection of data points for the training and testing sets is a significant disadvantage. This can lead to a great deal of unpredictability in a model's performance, which makes it challenging to draw reliable conclusions about how effectively the model generalizes to new data. Additionally, train-test separation frequently results in data waste, especially when dealing with little datasets. A portion of the data is set aside for testing, which reduces the quantity available for training models and raises the possibility of developing less reliable models that overfit the training set. Due to these restrictions, a different strategy is required. k-Fold Cross-Validation shows up as a potent remedy for these problems. Using this method, the dataset is split up into k folds, or subsets, and each fold is iteratively used for testing and training. Cross-validation yields a more reliable and representative estimate of a model's capacity for generalization by repeating this procedure k times and averaging the model's performance over these iterations. Most importantly, it guarantees effective data use, enabling every data point to be useful in both the training and testing stages. This capability is especially helpful when working with sparse data sets. In addition to improving stability, cross-validation is essential for model selection and hyperparameter tuning. It makes it possible to compare models or parameter settings with confidence, which fortifies the overall robustness of machine learning model assessment and development.

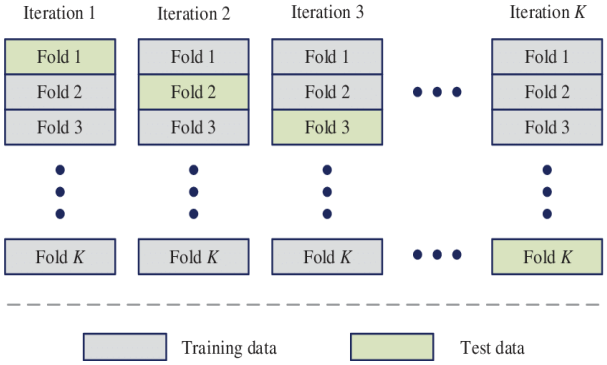

Figure 5. k-fold Cross Validation

Here are some limitations of CountVectorizer and how TfidfVectorizer can help mitigate them:

CountVectorizer Limitation: It considers the frequency of terms in a document but does not consider the importance of terms in the entire corpus.

TfidfVectorizer Solution: TfidfVectorizer (Term Frequency-Inverse Document Frequency) takes into account not only the term frequency in a document but also the inverse document frequency. This helps in giving higher weight to terms that are important within a specific document but not overly frequent across all documents.

Here are some limitations of train_test_split and how k-fold cross-validation can help:

Train-Test Split Limitation: Depending on the random split, you might end up with a portion of your data being exclusively used for training or testing, leading to a potential lack of information in one of the sets.

The performance of a model might be sensitive to a specific random split, and different splits may lead to different performance evaluations.

K-Fold Cross-Validation Solution: K-fold cross-validation divides the dataset into k folds and ensures that each fold is used for both training and testing. This helps in utilizing the entire dataset for training and testing, reducing the risk of biased performance estimates.

K-fold cross-validation averages the performance over multiple folds, providing a more stable and reliable estimate of a model's performance.

II. Literature Review

This article presents a comprehensive approach to spam Prediction using multiple classification methods, with a detailed analysis of their performance based on various evaluation metrics. Develop a spam Prediction model to enhance privacy and security by employing machine learning classification methods. The data used for training and testing the model was obtained from the UCI open repository. The dataset contains various attributes related to emails. Machine Learning Techniques: The following classification methods are employed:Naive Bayes, K*, J48, Random Forest.

Cross-validation: The proposed model undergoes cross-validation with 10 trials to achieve optimal performance. Evaluation Metrics: The performance of the model is assessed using metrics such as accuracy, recall, precision, and f1-measure[1].

Algorithms	Accuracy	Precision	Recall	F1-Measure
Naïve 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

Figure 6. Output Values for Spam Prediction Obtained from The Different Classifiers

The paper [2] demonstrates the effectiveness of using deep learning models, particularly a hybrid approach, for spam email classification, showcasing higher accuracy compared to traditional machine learning methods. The increasing volume of spam emails is a significant issue causing problems in communication systems. To present an approach that uses machine learning and deep neural networks for spam classification. The focus is on using various models, including Gaussian Naive Bayes (GNB), Convolution Neural Networks (CNN), Long Short-Term Memory (LSTM), and a customized model combining CNN and LSTM.

The Dataset is obtained from the open-source Spam assassin platform, consisting of around 6000 real email samples. The dataset contains both ham (not spam) and spam emails, with approximately 67% ham and 33% spam.

- The raw email body from the SpamAssassin dataset is used.
- For GNB, vectorization is performed, converting words into real numbers.
- For deep neural network models (LSTM, CNN, CNN-LSTM), tokenization is carried out, segregating the text into tokens (e.g., words) for natural language processing.
- The data is divided into a 70:30 ratio for training and testing purposes.
- Deep learning models (LSTM and CNN) achieved higher accuracy (98.27% and 98.68%, respectively) compared to the traditional machine learning model GNB (95.45% accuracy).
- A hybrid approach using CNN and LSTM [2] further improved the performance, reaching an accuracy of 98.68%.
- This task [3], involves the development of a predictive model for spam Prediction using various machine learning and neural network techniques. Data Collection:
- A spam-base dataset has been collected from the UCI machine learning repository.
- The dataset contains a total of 58 attributes, with 57 independent features and one dependent feature.

Attribute Types	Fifty five real and continuous attributes; two integer and continuous attributes; one nominal attribute
Number of Instances	4601
Number of spam instances	1813
Number of non-spam instances	2788
Used for	Classification
Characteristics of Dataset	Multivariate
Number of class in target feature	2 (0 denotes the email as non-spam and 1 represents the email as spam)

Figure 7. Information on spam-base dataset

Data Cleaning: Converting to Integer: This likely involves transforming certain data types to integers for uniformity and compatibility with the chosen algorithms.

Removing Outliers: Identifying and eliminating data points that deviate significantly from the rest of the dataset, as outliers can negatively impact model performance.

Feature Engineering:

Recursive Feature Elimination (RFE): A feature selection technique that recursively removes the least important features until the desired number of features is reached.

Heatmap: Visualization technique to analyze the correlation between different features in the dataset.

Chi-Square: A statistical test used to determine the independence of two categorical variables.

Data Splitting: The dataset is divided into an 80% training set and a 20% testing set.

Classification and Prediction models: Multinomial Naïve Bayes, K-Nearest Neighbor, Random Forest,

Gradient Boosting, Recurrent Neural Network (RNN), Gradient Descent, Artificial Neural Network (ANN).

Outlier Detection Technique	Deep Learning Algorithms	Loss Value	Accuracy
DBSCAN	Recurrent Neural Network (RNN)	0.6803	92.42 %
Isolation Forest	Recurrent Neural Network (RNN)	0.0450	99.28 %
DBSCAN	Gradient Descent	0.2568	89.42 %
Isolation Forest	Gradient Descent	0.0165	99.28 %
DBSCAN	Artificial Neural Network (ANN)	0.2469	91.42 %
Isolation Forest	Artificial Neural Network (ANN)	0.1333	97.95 %

Figure 8. Result analysis based on Deep Learning algorithms

The work[4], addresses the challenge of classifying emails as spam or non-spam. It employs a multi-step methodology:

1. ***Data Preprocessing:*** Cleaning the dataset by tokenization, stopword removal, and stemming.
2. ***Relationship Analysis:*** Assessing word relationships in email subjects and content, and scoring words based on entropy.
3. ***Feature Selection:*** Selecting the most informative words for email classification.
4. ***N-Grams:*** Generating N-grams (word sequences) from selected informative words.
5. ***TF-IDF Normalization:*** Reducing the high count of N-grams using TF-IDF.
6. ***CHI Square Feature Selection:*** Choosing top N-grams for classification.
7. ***Vocabulary Corpus:*** Constructing a vocabulary corpus for email classification.
8. ***Classification:*** Employing four classifiers, including Linear Support Vector Machine (LSVM), for email classification.

Results show that LSVM outperforms other classifiers with nearly 91% accuracy, high precision, and specificity. The research emphasizes word relationships in email content and subject as key to accurate classification, offering potential for further improvements in email filtering.

Experimental analysis is conducted on the Ling-Spam dataset, comprising 5000 emails divided into spam and ham categories. The dataset is split into training and testing sets using k-fold cross-validation. Confusion matrices are used to evaluate classifier performance, considering parameters such as true positives, true negatives, false positives, false negatives, specificity, precision, and prevalence. Preprocessing: Tokenization, stopword removal, and stemming are performed to clean the dataset. Relationship Analysis: The relationship between words occurring in the subject and email content is determined using entropy, which measures impurity in the dataset.

Feature Selection: Words are ranked based on their relationship score, and the top n words are selected as the most relevant and informative features for classification.

N-gram Generation: N-grams (unigram, bigram, and trigram) are created from the selected words to build a corpus for classification.

Normalization: The frequency of N-grams is normalized using TF-IDF(Term Frequency-Inverse Document Frequency) to handle high counts.

Feature Selection: Top n-grams are selected using the CHI Square method as a feature selector.

Classifier Implementation: Four classifiers, namely Linear Support Vector Machine, Multi-Naive Bayes, Decision Tree, and Random Forest, are employed for classification using the selected vocabulary corpus.

III. PROPOSED-FRAMEWORK

IDE - Google Collaborator / Python– Python 3

Spam Mail Prediction Data Source - https://www.kaggle.com/code/mohinurabdurahimova/spam-mail-prediction-machine-learning-project/input

File Size: 474 KB, Dataset Size:5572 rows × 2 columns, Applied Binary Classification Problemon Data

Table 1. Dataset Attributes

Sr.No	Feature	Description	Detailed Description
1	Category	Mail Type : Spam, Ham After Label Encoding Spam 0 Ham 1	Spam Mail - Fake or False Mail (Promotion Purse) Ham Mail - Not Spam Mail / True or legitimate Mail
2	Message	contains the actual text	It includes the text body of the communication.

- Importing Libraries
- Drive Mount and Data Collection
- Data Pre-processing Starts
- Check for Null Values :
- Data Pre-processing (Continued):
- CountVectorizer is used for Conversion of Text Data into Numerical Values
- Model Creation:
- Model Evaluation
- Text Pre-processing using TfidfVectorizer
- Cross Validation based Model Creation
- Model Evaluation

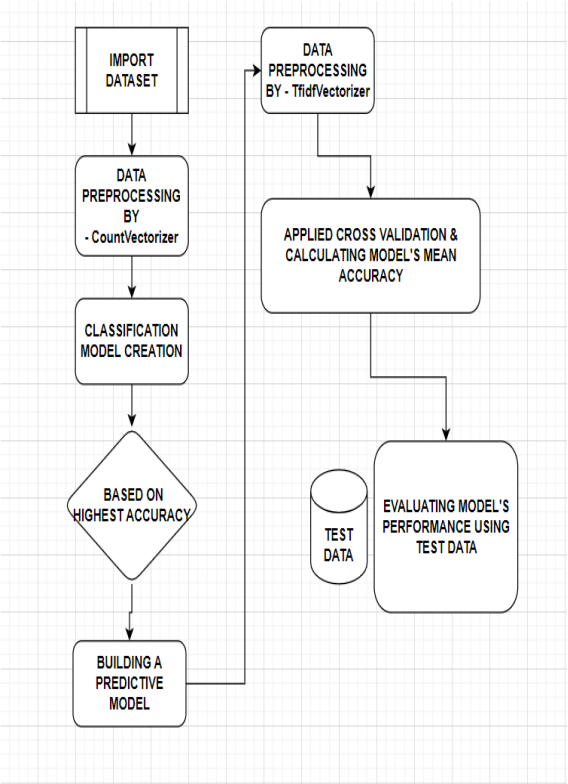

Figure 9. Proposed Work Flow

IV. RESULTS

This section presents the results and evaluation of the different classifiers with Train- Test and k-fold cross validation.

Table 2. Models Train Test and Cross Validation Mean Accuracies

ClassifierModels	Train Test Accuracy	Cross Validation, CV=10	
Logistic Regression	Accuracy - Train 0.997083239847431 Test 0.979372197309417	cv_score_LR [0.96594982 0.95698925 0.9497307 0.96768402 0.96050269 0.95332136 0.9443447 0.9551167 0.95152603 0.95332136]	mean_accuracy_LR 0.9558486644401973
Decision Tree	Accuracy – Train 1.0 Test 0.9721973094170404	cv_score_DT [0.98028674 0.97132616 0.96768402 0.97486535 0.97307002 0.97307002 0.96588869 0.96947935 0.96947935 0.97307002]	mean_accuracy_DT 0.9718219725487923
Random Forest	Accuracy - Train 1.0 Test 0.9775784753363229	cv_score_RF [0.99103943 0.97670251 0.98204668 0.97845601 0.97845601 0.97845601 0.98025135 0.97666068 0.97127469 0.97486535]	mean_accuracy_RF 0.9788208721839347
Support Vector Classifier	Accuracy - Train 0.9946152120260264 Test 0.9820627802690582	cv_score_SVC [0.9874552 0.96594982 0.97666068 0.98025135 0.97307002 0.96588869 0.97307002 0.97307002 0.97307002 0.97307002]	mean_accuracy_SVC 0.9741555825820608
KNeighbour	Accuracy - Train 0.9320170518285843 Test 0.9165919282511211	cv_score_KNN [0.90143369 0.91218638 0.91202873 0.90664273 0.90305206 0.90125673 0.9048474 0.91202873 0.90305206 0.91382406]	mean_accuracy_KNN 0.9070352567196258
GNB	Accuracy - Train 0.9452546555979359 Test 0.9165919282511211	cv_score_GNB [0.89247312 0.89247312 0.87253142 0.88150808 0.86175943 0.89048474 0.88150808 0.87253142 0.88868941 0.88150808]	Mean Accuracy GNB 0.881546688287871
XGB	Accuracy - Train 0.9952883105227731 Test 0.9766816143497757	cv_score_XGB [0.98566308 0.97132616 0.97486535 0.98025135 0.98025135 0.97307002 0.98025135 0.96947935 0.96768402 0.98384201]	Mean Accuracy: 0.9766684040848632

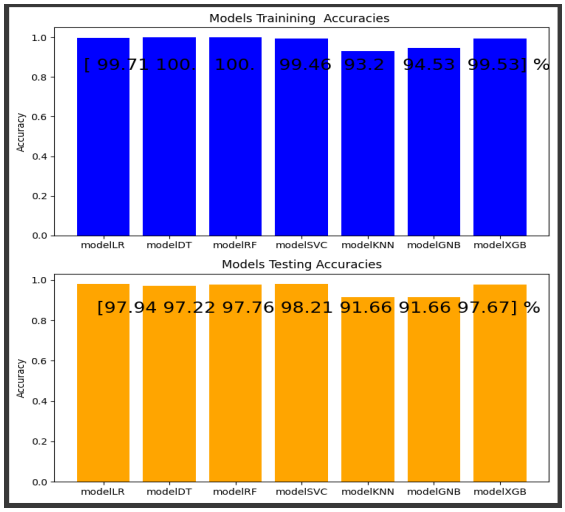

Figure 10. Train – Test Accuracy Graph

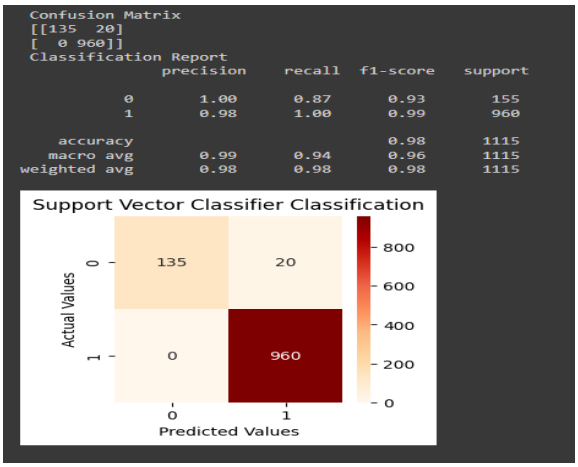

Figure 11. Based on Highest Accuracy Building a Predictive Model

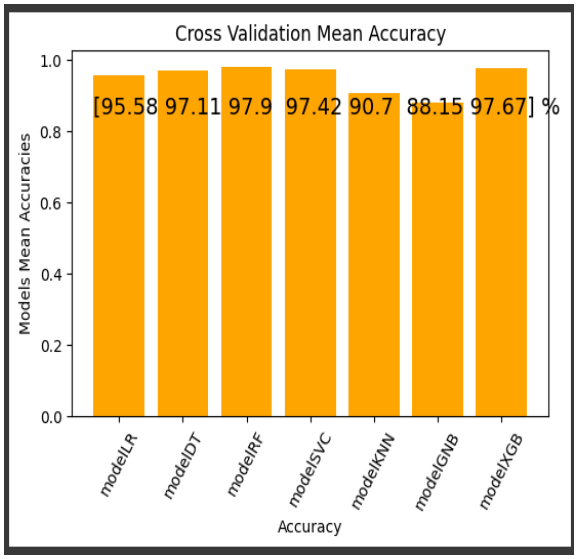

Figure 12. Models Cross Validation Mean Accuracy

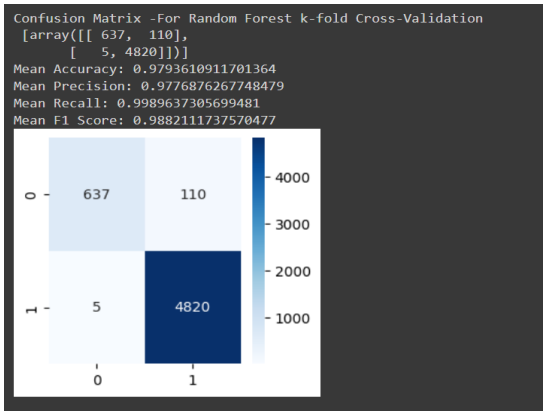

Figure 13. Building a Predictive Model based on Highest Cross Validation Mean Accuracy

V. Conclusion and Future Work

The following are the performance metrics of different classification models for the task: The goal of this job was to use machine learning to identify whether an email was spam (fake or fraudulent mail) or ham (legitimate mail). The dataset was acquired from a Kaggle source and has 5572 rows and 2 columns. Following preprocessing, the dataset included two main attributes: "Message," which contained the body of the communication, and "Category," which denoted the sort of mail (spam or ham). We carried out a thorough investigation, including data preprocessing and model assessment. Following the encoding of the "Category" feature, which assigned a value of 1 to ham and 0 to spam, we separated the dataset into training and testing sets. ML models developed and assessed using train and test sets. K-fold cross validation is applied throughout the entire data set and then evaluated to improve the model's performance.

- A strong test accuracy of 97.94% and an impressive train accuracy of 99.71% were attained by Logistic Regression. 95.58% was the average cross-validation accuracy.
- The decision tree demonstrated

exceptional test accuracy of 97.22% and flawless train accuracy of 100%. 97.18% was the average cross-validation accuracy.

- Random Forest also shown remarkable test accuracy of 97.76% and flawless train accuracy of 100%. Cross-validation accuracy on average was 97.88%.
- The Support Vector Classifier (SVC) achieved a remarkable test accuracy of 98.21% and a high train accuracy of 99.46%. Cross-validation accuracy was 97.42% on average.
- K Neighbours (KNN) achieved 91.66% test accuracy and 93.20% train accuracy. 90.70% was the mean cross-validation accuracy.
- The train accuracy of Gaussian Naïve Bayes (GNB) was 94.53%, and the test accuracy was 91.66%. Cross-validation accuracy on average was 88.15%.
- The test accuracy of 97.67% and the train accuracy of 99.53% were shown using Extreme Gradient Boosting (XGBoost). Cross-validation accuracy was 97.67% on average.

To construct the prediction model, the Random Forest model was chosen with a mean accuracy of 97.88% based on the highest cross-validation mean accuracy. With a mean accuracy of 97.88% and strong precision, recall, and F1 score, Random Forest's k-fold cross-validation demonstrated its strong performance in reliably classifying emails as spam and real.

All things considered; the Random Forest model turned out to be the most dependable option for categorising email messages due to its high cross-validation mean accuracy. It achieved an amazing balance between effectively learning from the training data and generalising to fresh, unseen email samples.

Future research in the field of spam mail prediction should concentrate on utilising cutting-edge technology such as deep learning models, especially transformer-based architectures, to improve the identification of complex spam patterns and context in email content. To further improve model performance, investigate ensemble approaches and hyperparameter adjustment. To quickly detect new threats, feature engineering, sophisticated text preprocessing, and real-time Prediction algorithms are essential. A comprehensive classification perspective can be obtained by integrating multimodal data, such as photographs or attachments, with email information.

Scalability and efficiency issues need to be resolved in order to successfully integrate these sophisticated models into email systems used in the real world. Email platforms with strong security awareness programmes can inform users about new strategies used by spammers. A more thorough understanding of model performance will also be possible with the adoption of a wider range of evaluation indicators. These developments are crucial for effectively thwarting the development of spam strategies and protecting users, as are deep learning, real-time Prediction, and multimodal data integration.

REFERENCES:

[1] Archana Saini, Kalpna Guleria, Shagun Sharma, "Machine Learning Approaches for an Automatic Email Spam Prediction", 2023 International Conference on Artificial Intelligence and Applications (ICAIA) Alliance Technology Conference (ATCON-1) | 978-1-6654-5627-2/23/$31.00 ©2023

IEEE | DOI: 10.1109/ICAIA57370. 2023.10169201.

[2] Junaid Mazhar Muhammad, Affan Bin Hasan, and Muhammad Farrukh, "Classification and Prediction of Spam Emails Based on AI Enabling Models Using Deep and Machine Learning Techniques", 2022 International Conference on Emerging Technologies in Electronics, Computing and Communication (ICETECC) | 978-1-6654-9087-0/22/$31.00 ©2022 IEEE |DOI: 10.1109/ICETECC56662. 2022.10069229.

[3] Fahima Hossain, Mohammed Nasir Uddin, Rajib Kumar Halder, "Analysis of Optimized Machine Learning and Deep Learning Techniques for Spam Prediction", 2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS) | 978-1-6654-4067-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/IEMTRONICS52119. 2021.9422508.

[4] Aakanksha Sharaff, Ulligaddala Srinivasarao, "Towards classification of email through selection of informative features", 2020 First International Conference on Power, Control and Computing Technologies (ICPC2T), 978-1-7281-4997-4/20/ $31.00 ©2020 IEEE.

[5] Lakshmana Phaneendra Maguluri, R. Ragupathy, Sita Rama Krishna Buddi, Vamshi Ponugoti, Tharun Sai Kalimil, "Adaptive Prediction of Spam Emails Using Bayesian Inference", Proceedings of the Third International Conference on Computing Methodologies and Communication (ICCMC 2019), IEEE Xplore Part Number: CFP19K25-ART; ISBN: 978-1-5386-7808-4.

[6] Simran Gibson, Biju Issac, Li Zhang, Seibu Mary Jacob, "Detecting Spam Email With Machine Learning Optimized With Bio-Inspired Metaheuristic Algorithms", January 2020, IEEE Access 8:187914-187932, DOI:10.1109/ACCESS. 2020.3030751.

[7] Naeem Ahmed, Rashid Amin, Hamza Aldabbas, Deepika Koundal, Bader Alouffi, and Tariq Shah, "Machine Learning Techniques for Spam Prediction in Email and IoT Platforms: Analysis and Research Challenges", Hindawi Security and Communication Networks Volume 2022, Article ID 1862888, 19 pages https://doi.org/10.1155/2022/ 1862888.

[8] B. Uday Reddy, S. Nagasai Tej, Md. Shoheb, Dr. Krishna Samalla, Y. Sreenivasulu, "Spam Mail Prediction Using Machine Learning", © 2023 IJCRT | Volume 11, Issue 5 May 2023 | ISSN: 2320-2882.

[9] M.A. Nivedha, S.Raja, "Prediction of Email Spam using Natural Language Processing Based Random Forest Approach", IJCSMC, Vol. 11, Issue. 2, February 2022, pg.7 – 22.

[10] Sneha Bobde, Sharvari Role, Lokesh Khadke, Tejas Shirude, Ms. Shital Kakad, "Email Spam Prediction Using Hybridization ofSVM and Random Forest", Inter national Journal of Research in Engineering and Science (IJRES), ISSN (Online): 2320-9364, ISSN (Print): 2320-9356, www.ijres.org Volume 11 Issue 7 ǁ

July 2023 ‖ PP. 188-193.

[11] Taher Jodiawala, Manav Bhagia, Prateek Duhoon, Shubhay Islaniya, Nivedeeta Mukherjee, Nutan Dolzake, "Intelligent Spam Mail Prediction System", International Research Journal of Engineering and Technology (IRJET), Volume: 10 Issue: 06 | Jun 2023 www.irjet.net p-ISSN: 2395-0072.

[12] Hardik N Patel, Shilpa Serasiya, "Machine Learning for Email Spam Messages Prediction", International Journal of Progressive Research in Engineering Management and Science (IJPREMS), Vol. 02, Issue 05, May-2022, pp : 44-48.

[13] Hari K.C., "Comparative Analysis and Prediction of Spam Email Classification Using Supervised Machine Learning Techniques", International Research Journal of Modernization in Engineering Technology and Science, Volume:03/ Issue:05/May-2021 Impact Factor-5.354 www.irjmets.com.

* * * * *