Volume: 10 Issue: VII Month of publication: July 2022

DOI: https://doi.org/10.22214/ijraset.2022.46072

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538*
*Volume 10 Issue VII July 2022- Available at www.ijraset.com*

# Performance Evaluation of Machine Learning Algorithms on Textual Datasets for Spam Email Classification

Shubham Mathur[1], Aakash Purohit[2]
*[1, 2]Department of C.S.E., J.N.V. University*

***Abstract**: Email is one of the most popular modes of communication we have today. Billions of emails are sent every day in our world but not every one of them is relevant or of importance. The irrelevant and unwanted emails are termed email spam. These spam emails are sent with many different targets that range from advertisement to data theft. Filtering these spam emails is very essential in order to keep the email space fluent in its functioning. Machine Learning algorithms are being extensively used in the classification of spam emails. This paper showcases the performance evaluation of some selected supervised Machine Learning algorithms namely Naive Bayes Classifier, Support Vector Machine, Random Forest, & XG-Boost for spam email classification on a combination of three different datasets. For feature extraction, both Bag of Words & TF-IDF models were used separately and performance with both of these approaches was also compared. The results showed that SVM performed better than all the other algorithms when trained with TF-IDF feature vectors. The performance metrics used were accuracy, precision, recall, and f1-score, along with the ROC curve.*
***Keywords: Machine learning Algorithm, Email Classification, Spam Filtering Naive Bayes, Support Vector Machine***

## I. INTRODUCTION

Email is one of the oldest modes of digital communication and it is being used since the 1970s. Today, billions of users send & receive emails every day because they are used for personal communication as well in a plethora of different fields such as academics, business, marketing, etc. Emails have remained relevant due to their results.

Emails remain one of the best sources of lead generation for businesses as compared to various social media platforms like Facebook & Twitter. As per Statista [1], around 306 billion emails were sent in 2020 and the number is expected to be around 347 billion by the end of 2023. This upward trend clearly shows that the popularity of email usage is surely going nowhere anytime soon.

The meaningful or genuine emails are often termed Ham while the unwanted emails that are generally sent in bulk to either annoy someone or to steal their data are often termed Spam. The majority of spam emails contain links to either phishing websites or to websites that are hosting malware.

Spammers lure people with lucrative emails to get them to interact with the spam. As per Statista, 45.1% of the total email traffic in March 2021 contained spam emails. As per Campaign Monitor [2], over 60 billion spam emails will be sent from 2019-to 2023. Spam emails are spread across various areas such as healthcare, online education/courses, free money/prize distributions, adult content, job offers, online gambling, etc to name a few. Classification of these spam emails is a very essential task these days as spam emails are increasing every day.

Machine Learning (ML) is a branch of Artificial Intelligence (AI) where computers are trained to learn explicitly without being programmed.

Among its subtypes, Supervised ML has been used popularly in email spam classification. Spamassassin, Lingspam, and a subset of the Enron-spam dataset [3] are used in this study and together they contain a large amount of textual data. The handling and processing of such textual data is a part of Natural Language Processing (NLP) which is a subfield of AI where computers are programmed to process and work with natural language data generated by humans. For transforming textual data into numerical features, two feature conversion approaches namely Bag of Words and TF-IDF are used in this research.

This study aims to evaluate the performance of some of the most commonly used supervised ML algorithms for classification including Multinomial Naive Bayes, Support Vector Machine, Random Forest, and XG-Boost with Bag of Words and TF-IDF feature vectors respectively.

## II. RELATED WORK

Spam emails were a big problem even before the advent of Machine Learning and NLP techniques. During that time some fixed rule-based techniques were used to classify incoming emails as spam or ham, such as checking the domain name of the sender in the header field of the email. If it belonged to a blacklisted domain then higher chance of it being spam. Other examples include checking the content of the email and scoring it with the help of a list containing the most commonly used spam words [4]. A score higher than a set threshold would classify the email as spam otherwise ham, etc [18]. However, with time these techniques quickly prove inefficient as spammers always bring new tricks into the scene to bypass all these rule checks. Generating more and more rules for each and every trick is not only time-consuming but also inefficient in terms of computational complexity. According to authors in [5], spam email is constantly changing and evolving [book]. To tackle this behaviour spam filters should also do the same. Machine learning algorithms combined with the power of NLP techniques to handle and pre-process textual data prove to be highly useful for spam email classification. Leading email servers, client software, and Internet Service Providers (ISPs) like Gmail, Yahoo mail, Outlook, etc. are using these techniques for spam filtering [6].

Email is mainly divided into two parts namely header and body, most of the data here are in form of text. Even in the case of a multimedia email, the subject line contains some text. The importance of textual data in spam email classification is thus very high. R. Miller and E.Y.A. Charles in [7], stresses the importance of analysis of email subject lines as they are first to be read by any recipient. However, Machine Learning algorithms can not work with textual datasets without feature selection and conversion processes. Bag of Words and Term Frequency-Inverse Document Frequency (TF-IDF) are two of the most popular techniques for these tasks. The majority of research works based on email spam classification or clustering used either one of these two techniques and then trained the learning algorithms [8]-[11].

In [12], the authors have presented a comprehensive review of email application areas, data sets used, feature space requirements, classification techniques, and usage of performance measures. In [13], the authors have compared four different types of Decision tree classifiers (ID3, J48, simple CART, and Alternating Decision Tree) in terms of classification accuracy in the WEKA environment. It was found that J48 outperforms the other classifiers in spam email classification.

## III. MACHINE LEARNING ALGORITHM

Supervised algorithms are mainly used for solving Classification and Regression problems[]. As Email spam filtering is a classification problem there is more focus on Supervised ML algorithms. In this research work, a performance evaluation of the four most commonly used Supervised ML algorithms is carried out which are discussed below:

### *A. Naive Bayes Classifier*

Naive Bayes is a general term for a family of classifiers which works on the principle of conditional probability given by Bayes theorem according to which the probability of occurring of an event A given the probability of other event B that has already occurred, is calculated as shown in Eq. (1).

$$P(A/B)=\frac{P(B/A).P(A)}{P(B)} \tag{1}$$

Where,
P(B|A) = Probability of B, given A is true,
P(A) = Probability of event A, and
P(B) = Probability of event B

The model is a probability table that is based on the feature values and updated according to the training data provided. When dealing with data having discrete features such as textual datasets, Multinomial Naive Bayes (MNB) is a suitable choice [14]. Due to this reason, they are popularly used for various text classification problems in NLP. The approach is called naive because it assumes the presence of absolute independence of features (which means the presence/occurrence of one feature is independent of the presence of any other feature) and each feature has an equal contribution to the result, which is not true most of the time especially when we are talking about real-world applications.

Despite this fact, Naive Bayes classifiers have shown considerable accuracy and speed for spam classification problems. The speed of these classifiers is very fast both in training and testing because of the simple calculations involved and thus they are used for making real-time predictions. They are easily scalable with large datasets and are resistant to noise and overfitting.

### *B. Support Vector Machine*

Support Vector Machines (SVM) are used for both classification and regression problems. The classification is done with the help of a decision boundary known as a Hyperplane. The data points are classified according to which side of the hyperplane they fall. The data points are plotted on an n-dimensional space where n denotes the number of features. The hyperplane depends on the value of n, for n=2 there will be a 2-dimensional space and the hyperplane will be a line. The given data can be linear or non-linear, when the data points are not linearly separable then SVM uses some complex mathematical functions to transform data points to make them separable. These functions are known as kernel functions and they covert low-dimensional input space to higher-dimensional space. Examples of these kernels are linear, polynomial, Radial Basis Function (RBF), etc. In this research work, the linear kernel is used as it provided the most accurate results according to the given dataset.

Choice of appropriate kernel function is necessary according to the given data. SVMs are very complex and take a long time in training but they provide fast and accurate predictions even with high-dimensional data. SVM works well with small datasets but performance degrades with large datasets, so they are not very scalable [15].

### *C. Random Forest*

Random Forests (RF) are supervised ML algorithms based on the concept of ensemble learning. In ensemble learning multiple learning models are combined to solve a problem, this ensures higher accuracy and fewer error rates in comparison to the scenario where a single learning model is solving a particular problem. Ensemble learning is divided into two categories called bagging and boosting, random forests use the bagging approach in which multiple Decision Trees (DT) are trained from different subsets of the given dataset. Thus, each tree would be constructed differently according to the training data provided to it. During the test phase, each DT will generate its own result and the final prediction of Random Forest will be decided by the majority voting process. For this to be accurate, all DTs in the forest must have a low correlation between them, otherwise, all of them will train and predict in the same way. In this research work, a Random Forest with 100 DTs is used with Entropy as the Attribute Selection Measure.

RF practically overcomes or mitigates most of the drawbacks of a single DT. They provide great accuracy, reduction in overfitting, and improved stability along with all advantages that a single DT has. However, due to multiple trees involved the complexity and computational costs of RF are much higher along with longer training times.

### *D. XGBoost*

Figures XGBoost is short for Extreme Gradient Boosting. It is an ensemble technique based on gradient boosting and is used for a variety of classification and regression problems and was introduced in [16]. In XGBoost the sequential learners are decision trees. Thus, in simple words, XGBoost implements gradient boosting on decision trees. In normal boosting, all input data points are assigned equal weight and are then fed to the first DT to predict results. The weight of incorrectly predicted variables is increased and then they are fed to the next DT and so on. So in this scenario, weights are updated for improving the results but gradient boosting is different. In gradient boosting the loss function is optimized, i.e. instead of weights, the loss (or residual) is passed to the next learner. XGBoost works on this principle to improve performance. XGBoost open-source library is available for various languages like C++, Python, R, Pearl, etc. It can run on a single machine as well as distributed processing frameworks.

## IV. PROPOSED METHODOLOGY

For this study, three different datasets were combined and the resulting dataset was pre-processed to remove null values, irrelevant characters, words, and other text as discussed in section 4.1. This pre-processed text was used to generate vocabulary followed by feature extraction phases and then finally these features were fed to ML algorithms for training as shown in Fig. 1

### *A. Data Collection and Pre-processing*

The three datasets used are Spamassassin, Lingspam, and a subset of Enron-spam corpus [17] which are publicly available for research. All three of them were in form of CSV files with index, body, and respective labels. Spam emails are labelled as 1 and Ham emails as 0.

Spamassassin contains 4150 ham emails and 1896 spam emails, Ling Spam contains 2172 ham emails and 433 spam emails, and lastly, Enron subset contains 5000 ham emails and 5000 spam emails. It can be seen that the former two are imbalanced and the last one is a balanced dataset. The combined dataset after removing null entries contains 11322 ham emails and 7328 spam emails.

Textual data is high-dimensional due to a lot of features [18].

Feature selection is essential for eliminating irrelevant features to avoid the problem of Curse of Dimensionality. In the combined dataset, the data pre-processing tasks such as tokenization, punctuation removal, stop-word removal, lower casing, and lemmatization were performed using Natural Language Toolkit (NLTK) library [19] in Python.

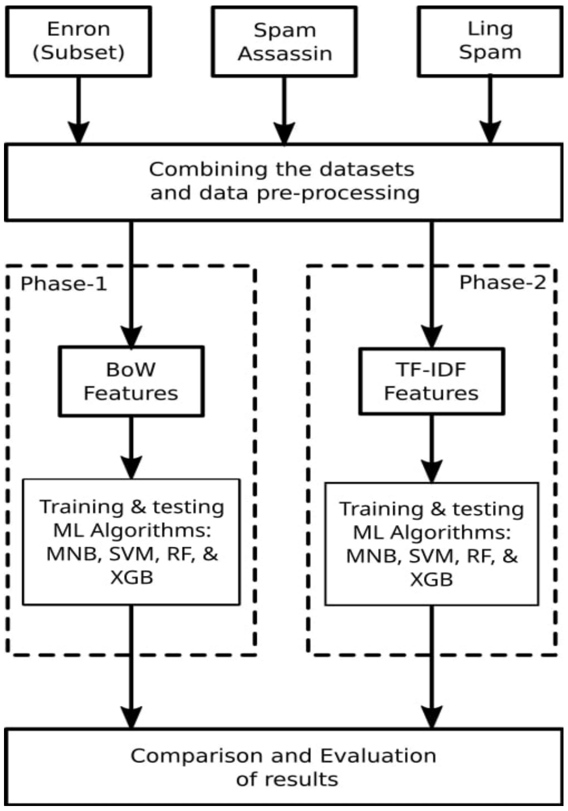
Enron (Subset)
Spam Assassin
Ling Spam
Combining the datasets and data pre-processing
Phase-1
Phase-2
BoW Features
TF-IDF Features
Training & testing ML Algorithms: MNB, SVM, RF, & XGB
Training & testing ML Algorithms: MNB, SVM, RF, & XGB
Comparison and Evaluation of results

Fig. 1 Proposed Methodology

## B. *Feature Extraction*

As ML algorithms can not directly work with raw text data, it is desirable to convert it into a suitable numeric form of fixed-length vectors. This is done using feature extraction and the two most popular techniques for this conversion are-

1) *Bag of Words (BOW):* A BoW model converts each text document into fixed-length vectors (equal to the size of vocabulary). Word count values of only those words which appear in a document are stored in the respective vector at suitable indices. These vectors are mostly sparse, especially in the case of large vocabularies. BoW representation does not contain any information about grammar or order of words as they appear in the original text and that is the reason for the simplicity of this approach. So as its name implies, BoW represents a document as a collection of words that occur in it.
2) *Term Frequency Inverse Document Frequency (TF-TDF):* A TF-IDF model is similar to BoW but the difference is in the way it re-scales the frequencies of each word. TF-IDF scores each word token by multiplying their TF and IDF values as explained below-

TF: Term Frequency is the number of times a term t occurs in document d (having N total terms) and is calculated using Eq. (2) [20].

$$tf\,(t,d) = \frac{n(t,d)}{N} \tag{2}$$

IDF: Inverse Document Frequency is a score that determines how rare a term t is in D (all documents) and is calculated using Eq. (3).

$$idf\,(t) = \log\left(\frac{D}{D_t}\right) \tag{3}$$

Where,
Dt = documents having term t.

Finally, for each term, the final score is calculated using Eq. (4).

$$tfidf(t) = tf(t,d) * idf(t) \tag{4}$$

These two approaches are suitable for feature extraction in email spam classification because we are only concerned with the occurrence of certain words and their importance when dealing with the textual contents of the email subject and body. Other NLP applications like sentiment analysis involve different vectorization approaches to convert the meanings of words along with them for accurate analysis.

## V. EVALUATION OF RESULTS

The entire dataset is spitted into 80% and 20% for training and testing respectively. In this study, both BoW and TF-IDF features are used separately with ML algorithms. In Phase-1 all algorithms were trained with BoW features extracted from the dataset and in Phase-2 all algorithms were trained with TF-IDF features extracted from the dataset. Then the results of all algorithms from Phase-1 and Phase-2 were evaluated. Performance evaluation of ML algorithms can be done in various ways, the most common one is overall accuracy. However, accuracy alone is not a sufficient performance metric for a classification algorithm especially when the training dataset is imbalanced. So, other metrics such as precision, recall, and f1-score are used [21]. All of these metrics are calculated using the following factors-

- *True Positive (TP):* Actual positive value predicted as positive
- *True Negative (TN):* Actual negative value predicted as negative
- *False Positive (FP):* Actual negative value predicted as positive
- *False Negative (FN):* Actual positive value predicted as negative

In the case of spam classification, email being spam is considered the positive case, and email being ham is the negative case. Now based on these factors the performance metrics namely Accuracy, Precision, Recall, and F1-score are calculated using Eqs. (5), (6), (7), and (8) respectively as explained below-

1) *Accuracy:* Ratio of correctly predicted cases to total cases

$$A = \frac{TP+TN}{TP+TN+FP+FN} \tag{5}$$

2) *Precision:* Ratio of TP to total positive predictions

$$P = \frac{TP}{TP+FP} \tag{6}$$

3) *Recall:* Ratio of TP to total actual positives

$$R = \frac{TP}{TP+FN} \tag{7}$$

4) *F1-score:* It is the harmonic mean of P and R

$$f1score = 2 \square \frac{P \square R}{P+R} \tag{8}$$

Table I contains performance metrics of all algorithms trained with BoW features in Phase 1 and Table II contains performance metrics of all algorithms trained with TF-IDF features in Phase 2.

Table I Performance of ML Algorithms with BOW features

| Algorithm | Accuracy | Precision | Recall | F1-score | ROC score |
|---|---|---|---|---|---|
| MNB | 0.95 | 0.95 | 0.93 | 0.94 | 0.95 |
| SVM | 0.96 | 0.92 | 0.96 | 0.94 | 0.96 |
| RF | 0.97 | 0.95 | 0.97 | 0.96 | 0.97 |
| XGB | 0.92 | 0.93 | 0.85 | 0.89 | 0.90 |

Table II Performance of ML Algorithms with TF-IDF features

| Algorithm | Accuracy | Precision | Recall | F1-score | ROC score |
|---|---|---|---|---|---|
| MNB | 0.94 | 0.99 | 0.85 | 0.92 | 0.93 |
| SVM | 0.97 | 0.95 | 0.98 | 0.96 | 0.97 |
| RF | 0.97 | 0.95 | 0.97 | 0.96 | 0.97 |
| XGB | 0.92 | 0.94 | 0.85 | 0.89 | 0.91 |

## A. *Confusion Matrix*

A confusion matrix is a combined representation of TP, TN, FP, and FN. It summarizes the classification performance of an ML algorithm in graphical or tabular format. Fig. 2, Fig. 3, Fig. 4, and Fig. 5 show the confusion matrices of MNB, SVM, RF, and XGB algorithms respectively (for both Phase-1 and Phase-2).

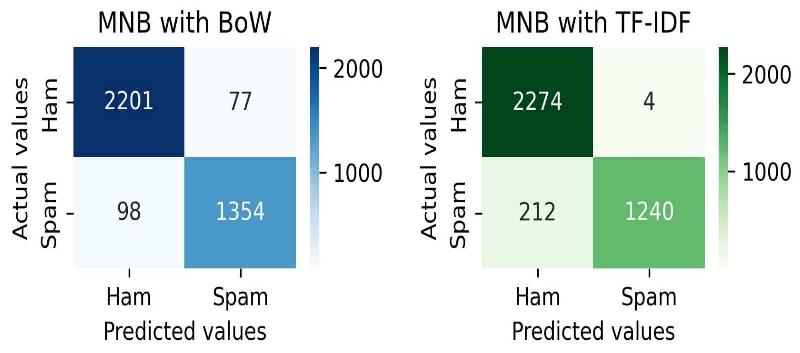

Fig. 2 MNB Confusion Matrices

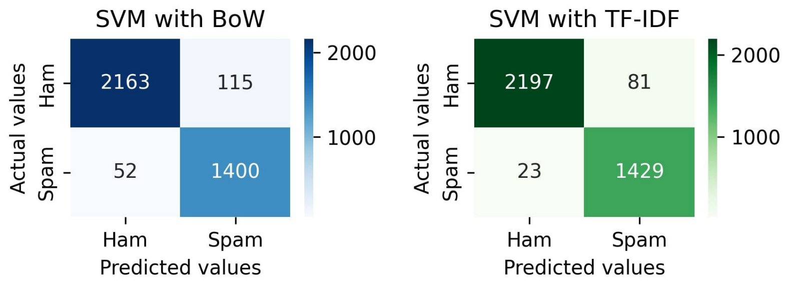

Fig. 3 SVM Confusion Matrices

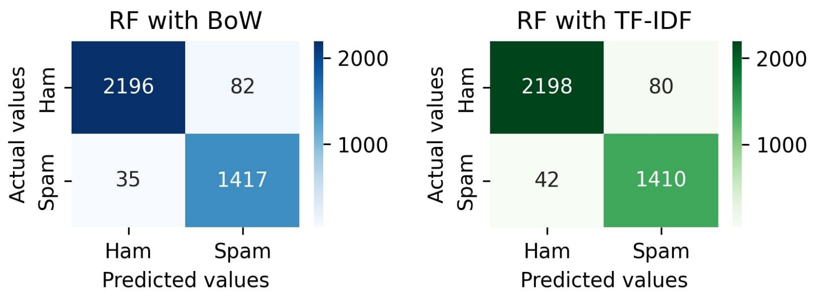

Fig. 4 RF Confusion Matrices

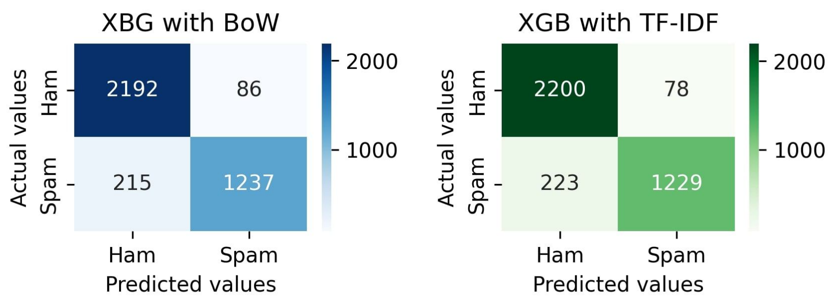

Fig. 5 XGB Confusion Matrices

*B. ROC Curve*

The Receiver Operator Characteristic curve is plotted on a graph between True Positive Rate (TPR) and False Positive Rate (FPR) on the y-axis and x-axis respectively. This graph shows the performance of a classification algorithm at all classification thresholds.

1) *TPR:* It is the ratio of TP to total actual positives (same as Recall, also called Sensitivity)
2) *FPR:* It is the ratio of FP to total actual negatives and is calculated using Eq. (9)

$$FPR = \frac{FP}{FP + TN} \tag{9}$$

Fig. 6, Fig. 7, Fig. 8, and Fig. 9 show ROC curves of MNB, SVM, RF, and XGB algorithms respectively (for both Phase-1 and Phase-2).

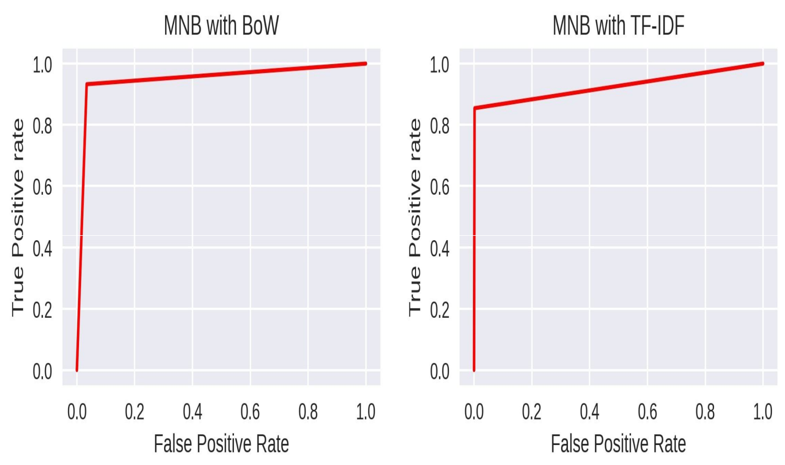

Fig. 6 MNB ROC Curves

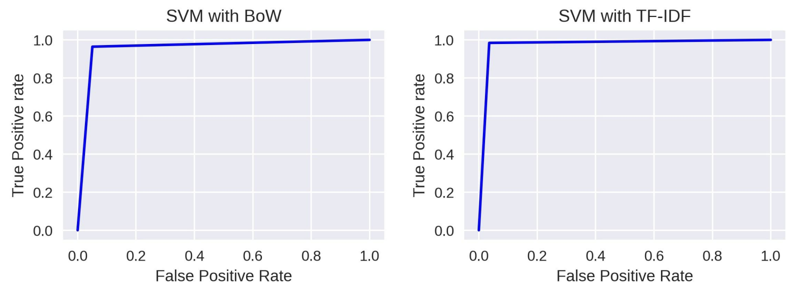

Fig. 7 SVM ROC Curves

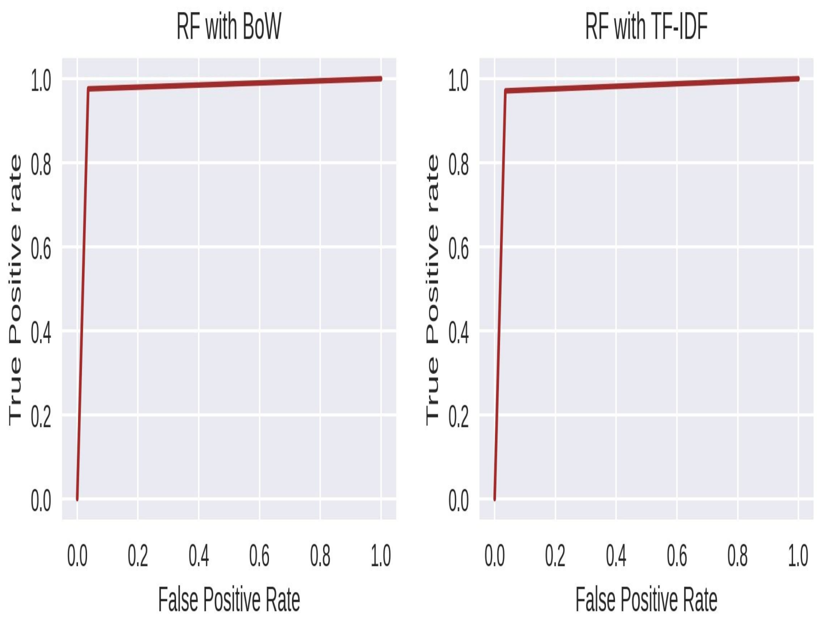

Fig. 8 RF ROC Curves

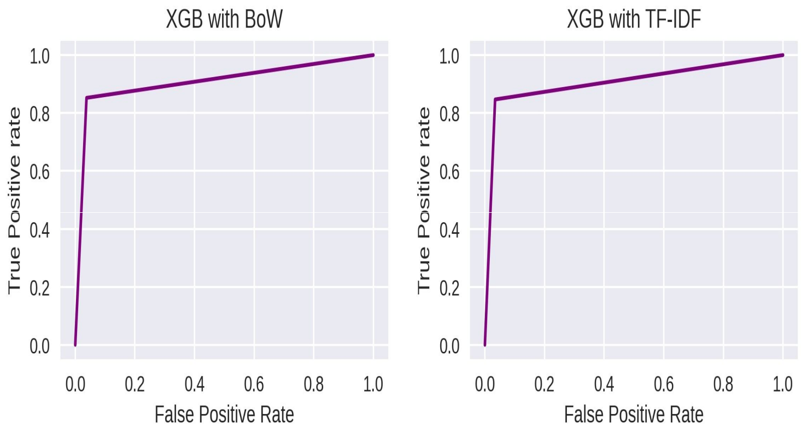

Fig. 9 XGB ROC Curves

## VI. CONCLUSIONS

In this study, three labelled datasets were combined to form a single dataset containing 18650 emails. The data was pre-processed to remove unnecessary contents and generate the vocabulary of useful tokens. Multinomial Naive Bayes, SVM, Random Forest, and XGBoost algorithms were trained with BoW and TF-IDF feature vectors separately and then the results were evaluated for both approaches. It was found that Multinomial Naive Bayes and KNN performed better with BoW vectors while SVM, Random Forest, and XGBoost performed better with TF-IDF vectors. Among all five algorithms, SVM performed the best with TF-IDF vectors and XGBoost scored the least in both phases. So, it can be seen that SVM still performs better than ensemble-based algorithms when it comes to text classification problems.

For future work, more emphasis will be on increasing the performance of all classifiers and reducing the number of False Positives as much as possible. This is because, in spam filtering, False Positives are not tolerable so the precision of the classifiers should be very high.

## REFERENCES

[1] (2022) Daily number of e-mails worldwide, Statista website. [Online]. Available: https://www.statista.com/statistics/456500/daily-number-ofe-mails-worldwide/

[2] (2022) Email usage statistics in 2021, Campaign monitor website. [Online]. Available: https://www.campaignmonitor.com/blog/email-marketing/email-usagestatistics-in-2019/

[3] B. Klimt and Y. Yang, "The Enron corpus: A new dataset for email classification research", in European Conference on Machine Learning, Springer, 2004, pp. 217–226.

[4] J. Alqatawna, H. Faris, K. Jaradat, M. Al-Zewairi, and O. Adwan, "Improving knowledge based spam detection methods: The effect of malicious related features in imbalance data distribution", in International Journal of Communications, Network and System Sciences, Vol. 8 No. 05, 2015, pp. 118.

[5] O. Saad, A. Darwish, and R. Faraj, "A survey of machine learning techniques for spam filtering", in International Journal of Computer Science and Network Security (IJCSNS), Vol. 12 No. 2, 2012, pp. 66.

[6] E. G. Dada, J. S. Bassi, H. Chiroma, A. O. Adetunmbi, and O. E. Ajibuwa, "Machine learning for email spam filtering: review, approaches and open research problems", Heliyon Vol. 5 No. 6, 2019.

[7] R. Miller and E. Charles, "A psychological based analysis of marketing email subject lines", in 2016 Sixteenth International Conference on Advances in ICT for Emerging Regions (ICTer), IEEE, 2016, pp. 58–65.

[8] M. Raza, N. D. Jayasinghe, and M. M. A. Muslam, "A comprehensive review on email spam classification using machine learning algorithms", in 2021 International Conference on Information Networking (ICOIN), IEEE, 2021, pp. 327–332.

[9] A. Radhakrishnan and V. Vaidhehi, "Email classification using machine learning algorithms", in International Journal of Engineering and Technology, Vol. 9 No. 2, 2017, pp. 335–340.

[10] I. Alsmadi and I. Alhami, "Clustering and classification of email contents", in Journal of King Saud University-Computer and Information Sciences, Vol. 27 No. 1, 2015, pp. 46–57.

[11] M. Mohamad and A. Selamat, "An evaluation on the efficiency of hybrid feature selection in spam email classification", in 2015 International Conference on Computer, Communications, and Control Technology (I4CT), IEEE, 2015, pp. 227–231.

[12] G. Mujtaba, L. Shuib, R. G. Raj, N. Majeed, and M. A. Al-Garadi, "Email classification research trends: review and open issues", IEEE Access 5, 2017, 9044–9064.

[13] A. K. Sharma and S. Sahni, "A comparative study of classification algorithms for spam email data analysis", in International Journal on Computer Science and Engineering, Vol. 3 No. 5, 2011, 1890–1895.

[14] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, "Multinomial naive bayes for text categorization revisited", in Australasian Joint Conference on Artificial Intelligence, Springer, 2004, pp. 488–499.

[15] S. Ray, "A quick review of machine learning algorithms", in 2019 International conference on machine learning, big data, cloud and parallel computing (COMITCon), IEEE, 2019, pp. 35–39.

[16] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting", in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016

[17] Spam filtering datasets, ACL (Association for Computational Linguistics) website. [Online]. Available: https://aclweb.org/aclwiki/Spam_filtering_datasets/

[18] A. Khan, B. Baharudin, L. H. Lee, and K. Khan, "A review of machine learning algorithms for text-documents classification", in Journal of advances in information technology, Vol. 1 No. 1, 2010, pp. 4–20.

[19] Nltk website. [Online]. Available: https://www.nltk.org/

[20] M. Ikonomakis, S. Kotsiantis, and V. Tampakas, "Text classification using machine learning techniques", in WSEAS transactions on computers, Vol. 4 No. 8, 2005, 966–974.

[21] Y. Alamlahi and A. Muthana, "An email modelling approach for neural network spam filtering to improve score-based anti-spam systems", International Journal of Computer Network and Information Security, Vol. 11 No. 12, 2018.