Spam Detection using Machine Learning and Natural Language Processing

Sri Ganesh Vathumalli

Instructor: Dr. S. Prince Mary M.E., Ph.D.,

Computer Science and Engineering

Sathyabama University, Tamil Nadu.

vathumallisriganesh@gmail.com

Manideep Sitaram Vattikuti

Instructor: Dr. S. Prince Mary M.E., Ph.D.,

Computer Science and Engineering

Sathyabama University, Tamil Nadu.

manideepsitaram143@gmail.com

Abstract: Spam email refers to any email which contains any kind of advertisement, unrelated or improper content. Today nearly 55% of all emails are categorized as spam. A lot of effort is being put in by service providers to reduce spam. In this paper, we present a new way to detect and classify spam emails using supervised machine learning models. These models in combination with natural language processing techniques called bag of words (BoW) and Term Frequency-inverse Document Frequency (TF-IDF) are used for classification. First, we present the Natural Language Processing(NLP) techniques used in this process and then we present various classifiers used. Then, we present our ensemble model and its working. Finally, we present results obtained from our model in terms of accuracy, precision, and f1 score.

Keywords: spam, NLP, ml, classification, entity, BOW, Bag-of-words, natural language, supervised classification.

I. INTRODUCTION

Nowadays email is one of the primary means of communication for billions of users which fits the requirement for some advertisers leading to an increase in spam emails. Spam emails are advertisements of various products, schemes, offers sometimes related to the user but most of the time unrelated to users. These emails are sent to the email addresses of millions of people acquired through various sources. Apart from annoying these emails can be used and are being used to spread various malware through hyperlinks and pop-up links can affect the host machine in multiple ways. Coming to advertisement emails, advertising, and offer mails are mainly directed for upping their sales. On the other hand, phishing emails are emails that look similar to advertising emails but the links and payment processes are created to extract information from users. Users unaware of this follow the hyperlinks to payment pages where users make payments providing the phishers with card information and personal information leading to identity theft and banking fraud sometimes serious crimes. The most dangerous type of email is malware-induced emails which are used by cybercriminals to gain access to personal computers and mobiles using emails. These emails look and feels similar to other spam email but contain malware embedded in them. Most of these emails contain an attachment that contains some kind of malware that infects the personal computer in various ways. Lots of instances occurred where a simple email blocked an entire network of computers. Mostly the user will be locked out of their machine and demanded a ransom to get into. Most of the email service providers deploy spam filters to protect their customers from spammers but those filters are limited constrained because some cases are present where important emails are being classified into spam directing the user to spam. No spam filter can classify the mail correctly all the time and increasing the filter intensity creates new problems.

In this paper, we address this problem by creating a tool that uses machine learning models and natural language processing techniques to classify an email as spam or not by subjecting it to various procedures. A list of words is

provided to the user that stands out from others in that email so that the user can decide if it is worthy to look at.

II. PAPER ORGANIZATION

The rest of this paper is formatted as follows. In section III we describe several research papers reviewed in the process. Section IV presents the system architecture and workflow of the tool. Section V describes various machine learning algorithms and their internal workings used. Section VI presents the experimentation procedure, results recorded, and comparisons with other research. Finally, in section VII we conclude.

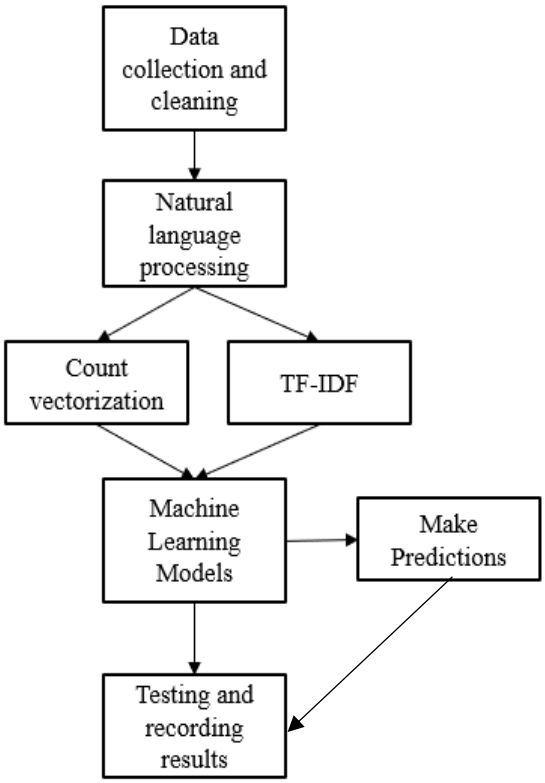
III. LITERATURE SURVEY

Spam classification is a problem that is neither new nor simple. A lot of research has been done and several effective methods have been proposed.

M. RAZA, N. D. Jayasinghe, and M. M. A. Muslam have analyzed various techniques for spam classification and concluded that naïve Bayes and support vector machines have higher accuracy than the rest, around 91% consistently. S. Gadde, A. Lakshmanarao, and S. Satyanarayana in their paper on spam detection concluded that the LSTM system resulted in higher accuracy of 98%. P. Sethi, V. Bhandari, and B. Kohli concluded that machine learning algorithms perform differently depending on the presence of different attributes. H. Karamollaoglu, İ. A. Dogru, and M. Dorterler performed spam classification on Turkish messages and emails using both naïve Bayes and support vector machines and concluded that the accuracies of both models measured around 90%. P. Navaney, G. Dubey, and A. Rana compared the efficiency of the SVM, naïve Bayes, and entropy method and the SVM had the highest accuracy (97.5%) compared to the other two models. S. Nandhini and J. Marseline K.S in their paper on the best model for spam detection it is concluded that random forest algorithm beats others in accuracy and KNN in building time. S. O. Olatunji concluded in her paper that while SVM outperforms ELM in terms of accuracy, the ELM beats the SVM in terms of speed. M. Gupta, A. Bakliwal, S. Agarwal, and P. Mehndiratta studied classical machine learning classifiers and concluded that convolutional neural network outperforms the classical machine learning methods by a small margin but take more time for classification. N. Kumar, S. Sonowal, and Nishant, in their paper, published that naïve Bayes produces the best results but has limitations due to class conditional classification and ensemble algorithms being better. T. Toma, S. Hassan, and M. Arifuzzaman studied various types of naïve Bayes algorithms and concluded that the multinomial naïve Bayes algorithm has better accuracy than the rest with an accuracy of 98%. F. Hossain, M. N. Uddin, and R. K. Halder in their study concluded that machine learning models perform better than deep learning models in spam classification and ensemble models outperform individual models in terms of accuracy and precision.

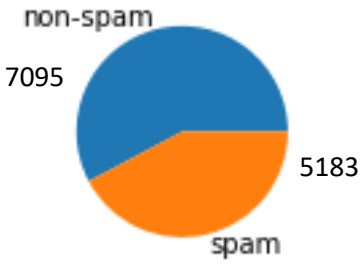
From various studies, we can take that for various types of data various models performs better. Naïve Bayes, random forest, SVM, logistic regression are some of the most used algorithms in spam detection and classification.

IV. RESEARCH METHODOLOGY



A) Data Collection and cleaning

A combination of two datasets containing around 12000 emails is used throughout the process. The data sets are combined to form a single dataset by removing any duplicates or null values. Collectively 5183 spam emails and 7095 ham emails are used for the process. The data used in the research are completely open-source and taken from Kaggle datasets. The data is further divided into training and testing datasets containing 80% of data and 20% of data respectively.



B) Natural Language Processing

Natural Language Processing is one of the major fields of study in the machine learning domain. This concerns the processing of natural language into machine language and creating natural language using machines. It is applied in most present-day electronics for implementing functions such as voice control and more. There are various domains within NLP such as sentiment analysis, speech recognition, text recognition, and many more.

In this paper, we used natural language processing(NLP) for text processing which includes various procedures. All the procedures are described in order of their application in text.

Tag removal: Removing all kinds of tags and unknown characters from text using regular expressions through the Regex library.

Sentencing, tokenization: Breaking down the text(email/SMS) into sentences and then into tokens(words). This process is done using the NLTK pre-processing library of python.

Stop word removal: Stop words such as of, a, be, … are removed using stopwords NLTK library of python.

Lemmatization: Words are converted into their base forms using lemmatization and pos-tagging. This process gives keywords through entity extraction. This process is done using chunking in regex and NLTK lemmatization.

Sentence formation: The lemmatized tokens are combined to form a sentence. This sentence is essentially a sentence converted into its base form and removing stop words. Then all the sentences are combined to form a text.

Feature vector formation: The texts are converted into feature vectors(numerical data) using the words present in all the texts combined. This process is done using the countvectorization of the NLTK library.

These procedures are applied to every email in data and on user emails.

Bag of words and term frequency-inverse document frequency is also used in this process

C) Bag of Words

Bag of words is a language model used mainly in text classification. A bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things:

- A vocabulary of known words.
- A measure of the presence of known words.

Below is a snippet of the first few lines of text from the book "A Tale of Two Cities" by Charles Dickens, taken from Project Gutenberg.

" It was the best of times,

it was the worst of times,

it was the age of wisdom,

it was the age of foolishness, "

The unique words here (ignoring case and punctuation) are:

[**"it", "was", "the", "best", "of", "times", "worst", "age", "wisdom", "foolishness"**]

The next step is to score the words in each document.

After scoring the four lines from the above stanza can be represented in vector form as

"It was the best of times" = **[1, 1, 1, 1, 1, 1, 0, 0, 0, 0]**

"it was the worst of times" = **[1, 1, 1, 0, 1, 1, 1, 0, 0, 0]**

"it was the age of wisdom" = **[1, 1, 1, 0, 1, 0, 0, 1, 1, 0]**

"it was the age of foolishness"= **[1, 1, 1, 0, 1, 0, 0, 1, 0, 1]**

This is the main process behind the bag of words but in reality the vocabulary even from a couple of documents is very large and words repeating frequently and important in nature are taken and remaining are removed during the text processing stage.

D) Term Frequency inverse document frequency (TF-IDF).

Term frequency-inverse document frequency is a measure of the originality of a word by comparing the number of times a word appears in a document with the number of documents the word appears in.

Terminology:

t – term(word)

d – document(set of words)

N – count of corpus

Corpus – total document set

i) Term Frequency

The number of times a word appears in a document is called term frequency.

$$tf(t, d) = count\ of\ t\ in\ d / number\ of\ words\ in\ d$$

ii) Document Frequency

Document frequency is the number of documents in which the word is present. We consider one instance of a word and it doesn't matter if the word is present multiple times.

$$df(t) = occurrence\ of\ t\ in\ documents$$

iii) Inverse Document Frequency

IDF is the inverse of document frequency which measures the informativeness of term t. all terms are considered equally important but certain terms such as (are, if, a, be, that, ..) provide little information about the document. The inverse document frequency factor diminishes the weight of terms that occur frequently and increases the weight of terms that occur rarely.

$$\boldsymbol{idf(t) = N/df}$$

Finally, the TF-IDF can be calculated by combining the term frequency and inverse document frequency.

$$\boldsymbol{tf\_idf(t,d) = tf(t,d) * \log(N/(df+1))}$$

the process can be explained using the following example:

Document 1 It is going to rain today.

Document 2 Today I am not going outside.

Document 3 I am going to watch the season premiere.

The Bag of words of the above sentences is

[going:3, to:2, today:2, i:2, am:2, it:1, is:1, rain:1]

Then finding the term frequency

Words	Document1	Document2	Document3
Going	0.16	0.16	0.12
To	0.16	0	0.12
Today	0.16	0.16	0
I	0	0.16	0.12
Am	0	0.16	0.12
It	0.16	0	0
Is	0.16	0	0
rain	0.16	0	0

Then finding the inverse document frequency

Words	IDF Value
Going	log(3/3)
To	log(3/2)
Today	log(3/2)
I	log(3/2)
Am	log(3/2)
It	log(3/1)
Is	log(3/1)
rain	log(3/1)

Applying the final equation the values of tf-idf becomes

Words/ documents	going	to	Today	i	am	if	it	rain
Document1	0	0.07	0.07	0	0	0.17	0.17	0.17
Document2	0	0	0.07	0.07	0.07	0	0	0
Document3	0	0.05	0	0.05	0.05	0	0	0

From the values, we can determine which word Is most important for which document.

E) Machine Learning Models

After completing the countvectorization and TF-IDF stages in the workflow the data is converted into vector form(numerical form) which is used for training and testing models. For our study various machine learning models are compared to determine which method is more suitable for this task. The models used for the study include Logistic Regression, Naïve Bayes, Random Forest Classifier, K Nearest Neighbors, and Support Vector Machine Classifier.

PROPOSED METHOD:

The unknown sample is converted into a numerical vector (BoW or TF-IDF). The language model is determined after the completion of the study. These vectors are given to machine learning models for making predictions.

The predictions made by models are grouped. These predictions are analyzed and the category that is predicted by the majority of the models is taken as the final prediction. Along with the category of the prediction the accuracy of the prediction to be true is also provided to the user.

V. MACHINE LEARNING

It is clear from the research that ensemble algorithms perform better than normal machine learning algorithms in both performance and time. Ensemble algorithms take results from multiple models and give the one which has the majority in terms of supporting algorithms.

I. Naïve Bayes Classifier

A naïve Bayes classifier is a supervised probabilistic machine learning model that is used for classification tasks. The main principle behind this model is the Bayes theorem.

Bayes Theorem:

Bayes theorem determines the conditional probability of event A given that event B has already occurred. Using Bayes theorem we can find the probability of event A happening, given that event B has occurred. Here, B is the evidence and A is the hypothesis. The condition applied here is that the features/predictors are independent of one another. The presence of one feature does not affect the presence of another. Hence called as naïve Bayes classifier.

$$\boldsymbol{P(A|B) = \frac{P(B|A)P(A)}{P(B)}}$$

P(A|B) – the probability of A happening given B.

P(B|A) – the probability of B happening given A.

P(A) – the probability of A happening.

P(B) – the probability of B happening.

Naïve Bayes classifiers are mostly used for text classification. The limitation of the Naïve Bayes model is that it treats every word in a text as independent and is equal in importance but every word cannot be treated equally important because articles and nouns are not the same when it comes to language. But due to its classification efficiency, this model is used in combination with other language processing techniques.

II. Random Forest Classifier

Random Forest classifier is a supervised ensemble algorithm. Ensemble algorithms are those which combines more than one algorithm of the same or different kind of algorithms. Random forest classifiers create a set of decision trees from a randomly selected subset of the training set.

Decision Tree:

The decision tree is a classification algorithm based completely on features. The tree repeatedly splits the data on a feature with the best information gain. This process continues until the information gained remains constant. Then the unknown data is evaluated feature by feature until categorized. Tree pruning techniques are used for improving accuracy and reducing the overfitting of data.

Several decision trees are created on subsets of data the result that was given by the majority of trees is considered as the final result. The number of trees to be created is determined based on accuracy and other metrics through iterative methods. Random forest classifiers are mainly used on condition-based data but it works for text if the text is converted into numerical form.

III. LOGISTIC REGRESSION

Logistic regression is a supervised classification algorithm based on regression. The probability of data to be in various categories is calculated and the category that gets major probability is selected. The probabilities are calculated using the Sigmoid function.

For example, let us take a problem where data has n features. We need to fit a line for the given data and this line can be represented by the equation

$$\boldsymbol{z = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3 \ldots . + b_n x_n}$$

here z = odds

generally, odds are calculated as

$$\boldsymbol{odds = \frac{p(event\ occurring)}{p(event\ not\ occurring)}}$$

Sigmoid Function:

A sigmoid function is a special form of logistic function hence the name logistic regression. The logarithm of odds is calculated and fed into the sigmoid function to get continuous probability ranging from 0 to 1.

The logarithm of odds can be calculated by

$$\boldsymbol{\log(odds) = dot(features, coefficients) + intercept}$$

and these log_odds are used in the sigmoid function to get probability.

$$\boldsymbol{h(z) = \frac{1}{1 + e^{-z}}}$$

The output of the sigmoid function is an integer in the range 0 to 1 which is used to determine which class the sample belongs to. Generally, 0.5 is considered as the limit below which it is considered a NO, and 0.5 or higher will be considered a YES. But the border can be adjusted based on the requirement.

IV. K-NEAREST NEIGHBORS (KNN)

K-Nearest Neighbors is a supervised machine learning classification algorithm. All the data points are assumed to be in an n-dimensional space. And then based on neighbors the category of current data is determined based on the majority.

Euclidian distance is used to determine the distance between points. The distance between 2 points is calculated as

$$\boldsymbol{d = \sqrt{(x2 - x1)^2 + (y2 - y1)^2}}$$

The distances between the unknown point and all the others are calculated. Depending on the K provided k closest neighbors are determined. The category to which the majority of the neighbors belong is selected as the unknown data category.

If the data contains up to 3 features then the plot can be visualized. It is fairly slow compared to other distance-based algorithms such as SVM as it needs to determine the distance to all points to get the closest neighbors to the given point.

VI. SUPPORT VECTOR MACHINES (SVM)

It is a supervised machine learning model for classification. Decision boundaries are drawn between various categories and based on which side the point falls to the boundary the category is determined.

Support Vectors:

The vectors that are closer to the boundary are called support vectors. If there are n categories then there will be n+1 support vectors. Instead of points, these are called vectors because they are assumed to be starting from the origin.

The distance between the support vectors is called margin. We want our margin to be as wide as possible because it yields better results.

There are three types of boundaries used by SVM to create boundaries.

Linear: used if the data is linearly separable.

Poly: used if data is not separable. It creates any data into 3-dimensional data.

Radial: this is the default kernel used in SVM. It converts any data into infinite-dimensional data.

If the data is 2-dimensional then the boundaries are lines. If the data is 3-dimensional then the boundaries are planes. If the data categories are more than 3 then boundaries are called hyperplanes.

An SVM mainly depends on the decision boundaries for predictions. It doesn't compare the data to all other data to get the prediction due to this SVM's tend to be quick with predictions.

VI. EXPERIMENTATION & RESULTS

The process goes like data collection and processing then natural language processing and then vectorization then machine learning.

The data is collected, cleaned, and then subjected to natural language processing techniques specified in section IV. Then the cleaned data is converted into vectors using Bag of Words and TF-IDF methods which goes like...

The Data is split into Training data and Testing Data in an 80-20 split ratio. The training and testing data is converted into Bag-of-Words vectors and TF-IDF vectors as described in section IV.

There are several metrics to evaluate the models but the F1 score is a better metric compared to precision and accuracy.

F1 score:

The F-score, also called the F1-score, is a measure of a model's accuracy on a dataset. It is used to evaluate binary classification systems, which classify examples into 'positive' or 'negative. The F1 score is the harmonic mean of precision and recall.

$$\boldsymbol{F1} = \mathbf{2} * \frac{(\boldsymbol{precision} * \boldsymbol{recall})}{(\boldsymbol{precision} + \boldsymbol{recall})}$$

Here,

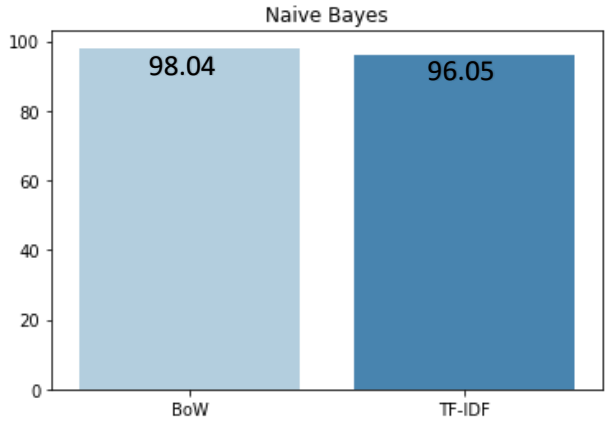
$$precision = \frac{true\ positives}{true\ positives + false\ positives}$$

$$recall = \frac{true\ positives}{true\ positives + false\ negatives}$$

Here F1 score is used because it takes both precision and recall into account and works well with imbalanced data.

Naïve Bayes:

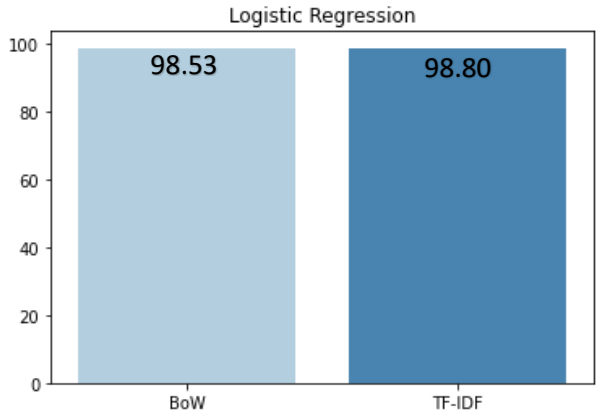
Two models, one for Bow and one for TF-IDF are created and trained using respective training vectors and training labels. Then the respective testing vectors and labels are used to get the score for the model. The scores for Bag-of-Words and TF-IDF are visualized below.



The scores for the Bow model and TF-IDF models are 98.04 and 96.05 respectively.

Logistic Regression:

Two models are created following the same procedure used for naïve Bayes models and then tested the results obtained are visualized below.



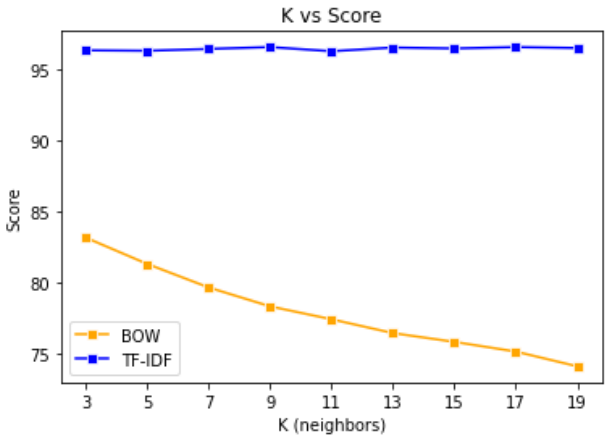
The scores for BoW and TF-IDF models are 98.53 and 98.80 respectively.

K-Nearest Neighbors:

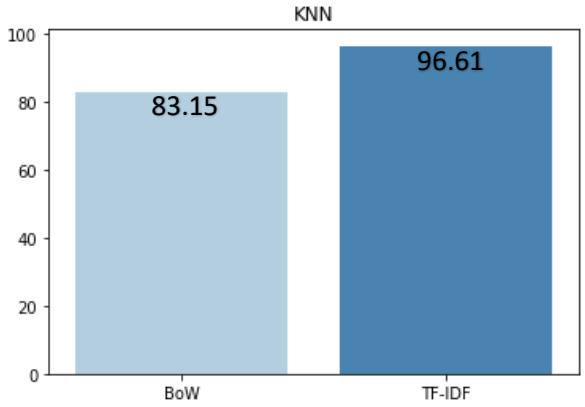
Similar to the above models the models are created and trained using respective vectors and labels. But in addition to the data, the number of neighbors to be considered should also be provided.

Using Iterative Method K =3 (no of Neighbors) provided the best results for the BoW model and K = 9 provided the best results for the TF-IDF model.

The K vs Scores for both models are converted into visual form for better understanding.



The scores for BoW and TF-IDF models are visualized below.

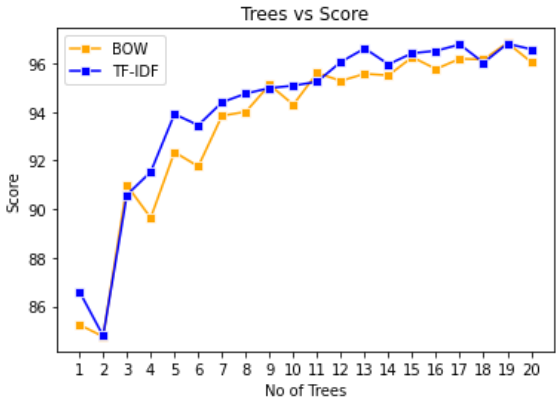


The scores for BoW and TF-IDF models are 83.15 and 96.61 respectively.

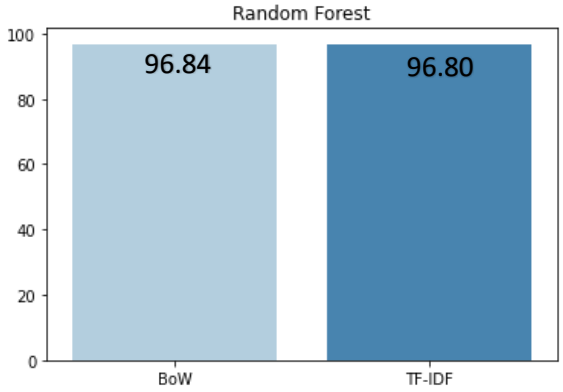
Random Forest:

Similar to previous algorithms two models are created and trained using respective training vectors and training labels. But the number of trees to be used for forest has to be provided.

Using the Iterative method best value for the number of trees is determined. From the results, it is clear that 19 estimators provide the best score for both the BoW and TF-IDF models. The no of tress and scores for both models are visualized.



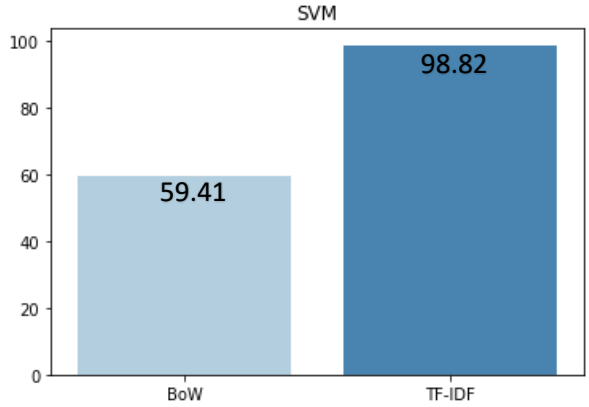
The scores for BoW and TF-IDF models are visualized below.



The scores obtained for BoW and TF-IDF are 96.84 and 96.80 respectively.

Support Vector Machines:

Finally, two SVM models, one for BoW and one for TF-IDF are created and then trained using respective training vectors and labels. Then tested using testing vectors and labels.



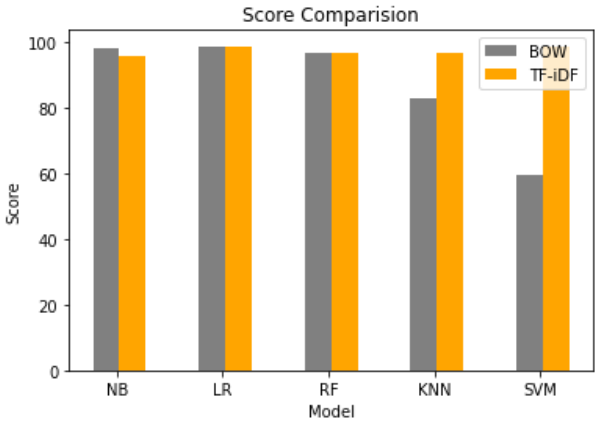
The scores for BoW and TF-IDF models are 59.41 and 98.82 respectively.

Combining the results from all the models gives a complete picture of both models.

Models	Bag of Words	TF-IDF
Naïve Bayes	98.04	96.05
Logistic Regression	98.53	98.80
Random Forest	96.84	96.80
KNN	83.15	96.61
SVM	59.41	98.82

Finally, to compare the results, the scores for each model using both bow and TF-IDF are converted to visual form.

The results from both models in visual form are presented side by side for easier comparison.



From the results obtained it is clear that TF-IDF outperforms the BoW model in every algorithm used.

PROPOSED SYSTEM

Since TF-IDF is better than BoW, it is used as the language model for our proposed system.

In our proposed system we combine all the models and make them into one. It takes an unknown point and feeds it into every model to get predictions. Then it takes these predictions, finds the category which was predicted by the majority of the models, and finalizes it.

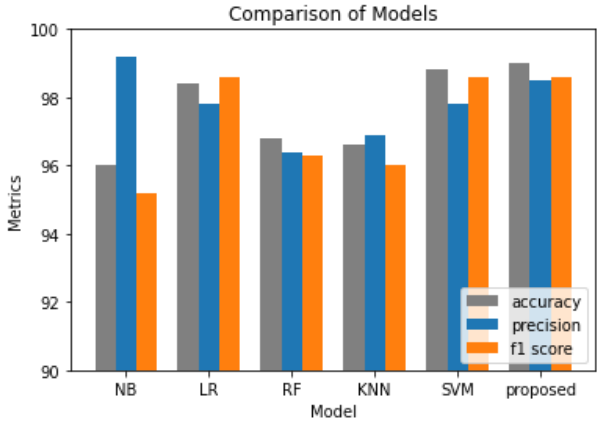
To determine which model is effective we used three metrics Accuracy, Precision, and F1score. In the earlier system, we used only the F1 Score because we were not determining which model is best but which language model is best suited for classification.

Metric / Model	Accuracy	Precision	F1 Score
Naïve Bayes	96.0	99.2	95.2
Logistic Regression	98.4	97.8	98.6
Random forest	96.8	96.4	96.3
KNN	96.6	96.9	96.0
SVM	98.8	97.8	98.6
Proposed model	99.0	98.5	98.6

The color RED indicates that the value is lower than the proposed model and GREEN indicates equal or higher.

Here we can observe that our proposed model outperforms almost every other model in every metric. Only one model(naïve Bayes) has slightly higher accuracy than our model but it is considerably lagging in other metrics.

The results are visually presented below for easier understanding and comparison.



The accuracy, precision, and f1 score for every model are calculated using testing vectors and testing labels.

VII. CONCLUSION

In this paper, we proposed a new method to classify spam using multiple machine learning models. We started with two language models BoW and TF-IDF but the TF-IDF language model proved to be more efficient in the classification of spam. Using TF-IDF a model has been created using the ensemble method. This Proposed model outperformed almost all of the other models in every metric proving that Ensemble algorithms perform better in classification than individual models. This proves that NLP can be effectively used to tackle other problems and provide better solutions.

REFERENCES

J. Fattahi and M. Mejri, "SpaML: a Bimodal Ensemble Learning Spam Detector based on NLP Techniques," 2021 IEEE 5th International Conference on Cryptography, Security and Privacy (CSP), 2021, pp. 107-112, DOI: 10.1109/CSP51677.2021.9357595.

M. RAZA, N. D. Jayasinghe and M. M. A. Muslam, "A Comprehensive Review on Email Spam Classification using Machine Learning Algorithms," 2021 International Conference on Information Networking (ICOIN), 2021, pp. 327-332, DOI: 10.1109/ICOIN50884.2021.9334020.

S. Gadde, A. Lakshmanarao, and S. Satyanarayana, "SMS Spam Detection using Machine Learning and Deep Learning Techniques," 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), 2021, pp. 358-362, DOI: 10.1109/ICACCS51430.2021.9441783.

P. Sethi, V. Bhandari and B. Kohli, "SMS spam detection and comparison of various machine learning algorithms," 2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN), 2017, pp. 28-31, DOI: 10.1109/IC3TSN.2017.8284445.

H. Karamollaoglu, İ. A. Dogru, and M. Dorterler, "Detection of Spam E-mails with Machine Learning Methods," 2018 Innovations in Intelligent Systems and Applications Conference (ASYU), 2018, pp. 1-5, DOI: 10.1109/ASYU.2018.8554014.

P. Navaney, G. Dubey and A. Rana, "SMS Spam Filtering Using Supervised Machine Learning Algorithms," 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2018, pp. 43-48, DOI: 10.1109/CONFLUENCE.2018.8442564.

S. Nandhini and J. Marseline K.S., "Performance Evaluation of Machine Learning Algorithms for Email Spam Detection," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020, pp. 1-4, DOI: 10.1109/ic-ETITE47903.2020.312.

S. O. Olatunji, "Extreme Learning Machines and Support Vector Machines models for email spam detection," 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE), 2017, pp. 1-6, DOI: 10.1109/CCECE.2017.7946806.

M. Gupta, A. Bakliwal, S. Agarwal, and P. Mehndiratta, "A Comparative Study of Spam SMS Detection Using Machine Learning Classifiers," 2018 Eleventh International Conference on Contemporary Computing (IC3), 2018, pp. 1-7, DOI: 10.1109/IC3.2018.8530469.

N. Kumar, S. Sonowal and Nishant, "Email Spam Detection Using Machine Learning Algorithms," 2020 Second International Conference on Inventive Research in Computing Applications (CIRCA), 2020, pp. 108-113, DOI: 10.1109/ICIRCA48905.2020.9183098.

T. Toma, S. Hassan, and M. Arifuzzaman, "An Analysis of Supervised Machine Learning Algorithms for Spam Email Detection," 2021 International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI), 2021, pp. 1-5, DOI: 10.1109/ACMI53878.2021.9528108.

F. Hossain, M. N. Uddin, and R. K. Halder, "Analysis of Optimized Machine Learning and Deep Learning Techniques for Spam Detection," 2021 IEEE International IoT, Electronics and Mechatronics Conference (IEMTRONICS), 2021, pp. 1-7, DOI: 10.1109/IEMTRONICS52119.2021.9422508.