Enhancement of Twitter Spam Detection Using Naïve Bayes Algorithm

Jan Michael I. Carpo [1], Joshua E. Adelante [1], Dominic R. Del Castillo [1], Mark Christopher R. Blanco [1], Ariel M. Sison [1]

[1]Student, Computer Science Department, College of Engineering - Pamantasan ng Lungsod ng Maynila, University of the City of Manila, Intramuros, Manila 1002, Philippines.

Corresponding Author: jmicarpo2018@plm.edu.ph

Abstract: - The Naive Bayes classification algorithm is a widely used method suitable for both binary and multiclass classification tasks. Unlike numerical input variables, Naive Bayes performs well when dealing with categorical input variables. It is commonly employed in applications such as sentiment analysis, spam filtering, and recommendation systems. One advantage of Naive Bayes is its ability to make predictions and anticipate data based on past outcomes. It is known for its simplicity and efficiency, requiring less training data compared to other models. However, a major limitation is its assumption of independent predictors, which may not hold true in real-world scenarios. Despite this drawback, Naive Bayes exhibits better performance and offers a wider range of predictions when researchers incorporate improvements and advancements. This makes it a suitable choice for multi-class prediction problems. Researchers have conducted extensive testing and simulations, leading to significant advancements in the algorithm's performance, including improvements in vocabulary, accuracy, and speed. Nonetheless, there are still unresolved challenges in automatic text processing, particularly in the domain of spam identification in social networks. Ongoing research aims to overcome these challenges and further enhance the capabilities of Naive Bayes and automatic text processing techniques.

Key Words: — ***Twitter, Spam, Naïve Bayes, Tweet, Machine Learning, Filtering System.***

I. INTRODUCTION

Online social networks like Facebook, LinkedIn, and Twitter have gained immense popularity as platforms for communication and collaboration. However, the rise of spam, including counterfeit comments and unsolicited messages, poses significant challenges. Spam messages aim to spread misinformation, perpetrate fraud, and advertise products or services, leading to resource consumption and prolonged communication time [1]. To address these issues, researchers and platform managers are implementing measures to prevent detrimental outcomes and enhance the online social networking experience. Twitter, being one of the largest social networks, faces complexities in spam detection due to the unique structure of tweets. Detecting spam on social networks, particularly on Twitter, requires leveraging data mining and machine learning techniques, considering not only in-text features but also the social network's structure, user information, and message relationships [2]. By incorporating these aspects, it becomes possible to enhance the accuracy and efficiency of spam detection and mitigate the negative impacts of spam on online social networking.

Manuscript revised June 16, 2023; accepted June 17, 2023. Date of publication June 19, 2023.

This paper available online at www.ijprse.com

ISSN (Online): 2582-7898; SJIF: 5.59

II. RELATED LITERATURE

2.1 Naive Bayes Classifier and its role in implementing Spam Detection

Using a directed social graph model for Twitter spam detection has resulted in high precision rates. The paper presents a graph and content-based model that follows Twitter's spam policy, and Naive Bayes is still considered the most precise algorithm

for spam detection. [3] In addition, some have identified nine key attributes that can help determine whether a user account is real or fake, including Profile Created, Favorite Count, Follower Count, Following Count, Geo Enabled, Follower Rate, Following Rate, Follower Following Ratio, and Verified. [4] Although the verification attribute has changed since the paper's publication, the other categories remain helpful for identifying fake accounts. The initial results showed an 80% accuracy rate in identifying fake accounts, which can be improved with a larger dataset. [5] Researchers used the Naïve Bayes algorithm with pre-processing stages (e.g., case folding, cleaning, tokenizing, and stemming) to classify comments in social media as spam or non-spam. With similar results of 80% accuracy and a precision of 0.72, the algorithm has successfully classified comments on social media.

Studies have demonstrated that fake tweets are distributed as frequently as genuine tweets. Automated accounts, or bots, are instrumental in hastening the spread of spam, with human users contributing to its proliferation. Several studies have been conducted to detect spam and spammers, which will be expounded upon in the ensuing discussion.

The research conducted by Wang is considered a pioneering effort in detecting Twitter spam by analyzing the relationships between users. This approach uses a direct graph model to identify follower relationships between users and detects spam by extracting textual features. Generally, previous research on Twitter spam detection can be categorized into the analysis of social network graph structures, [6] analysis of text structure, and the extraction of patterns are two methods commonly used to detect spam in textual data [7], analysis involves looking at user profile details and applied URLs [8]. Furthermore, another approach to detecting spam on Twitter involves analyzing the interactive behavior of users [9]

With other studies which employ Twitter data for five purposes: real-time filtering, scalability, precise decision-making, retraining models with new data, and text-independent classification. The dataset used for the study comprises 500,000 records, of which 400,000 are utilized for training purposes and 100,000 for testing. [10] The training dataset is tested using one-to-one (half regular and half spam), one-to-four, and one-to-ten combinations. The achieved precision is 91%, indicating high accuracy. The significant advantage of this study is that it is a real-time system that can filter data with a short latency and is scalable for handling extensive data. However, the main disadvantage of the study is that the entire dataset is unavailable, and it only uses one algorithm. Moreover, since the number of spammers in the given dataset is low compared to regular users, the model's evaluation needs to be completed as it only considers normal users while ignoring spammer users. To address this issue, the proposed method balances the spam and non-spam classes in the dataset, maintaining a corresponding ratio, which enhances the model's effectiveness.

In their 2017 paper, Liu and colleagues proposed a novel approach to detecting spam on Twitter that emphasizes the importance of data structure. They observed that imbalanced datasets, where spam and non-spam data are not represented proportionally, can lead to errors in machine learning classifiers. While spam only accounts for 5-10% of posts on Twitter, some datasets classify spam and non-spam data in a 50:50 ratio, resulting in lower precision.

Their three-step process involves data sampling using replacement, non-replacement, and fuzzy sampling. Next, each sampling technique uses machine learning classifiers such as support vector machine classifiers, decision trees, and simple Bayesian models to classify the samples as spam or non-spam. In the final step, a majority vote rule is used to decide whether the message is spam. While this approach is powerful in terms of the classifiers used, it relies solely on statistical information from the text itself as the feature for the classifiers, which can lead to lower precision [11].

Another study examines various user-based and content-based features that differentiate spammers from legitimate users and employs them for spam detection. Twitter's API methods retrieve information about active users, their followers and followings, and their 100 most recent tweets. The detection process is then evaluated based on these features. Results indicate that the Random Forest classifier outperforms the other classifiers and achieves a precision and F-measure of 95.7% in spam detection. [12]

In further development, spam messages can be a nuisance and threaten social media users' privacy and security [13]. To address this problem, researchers have proposed a methodology based on machine learning algorithms to detect and avoid spam messages on social media platforms like Twitter. The proposed approach uses a set of content-based features to develop the spam detection model, and two machine learning algorithms, support vector machine (SVM) and Naive Bayes classification

algorithm, are utilized for classification. The study results show that Naive Bayes performs better than SVM, with a performance measure of over 92% for most datasets without cross-validation. When cross-validation is used, the NB classifier achieves a performance measure of over 93%, compared to the SVM classifier. These findings suggest that the proposed methodology effectively detects and prevents spam messages on social media and that the Naive Bayes classifier is a good choice for this task.

Table 2.4.1 displays and contrasts the various types of previous systems and their unique details like descriptions and limits.

Table 2.4.1 Comparison of Previous Systems

Previous System	**Interpretation**	**Limitation**
Stop Words	Stop words are occurring words that appear in the text (such as articles, pronouns, prepositions, and conjunctions) that are often removed from the text during natural language processing tasks to improve efficiency and focus on more meaningful content.	Accuracy of knowledge models Unexpected outcomes
Vocabulary density and richness	It is an indicator of how many different kinds of words are employed in a text or language.	Can not filter a broader range of vocabulary New words can pass thru the filter.
Feature Extraction	The process of selecting and transforming raw data into a reduced set of representative features enables efficient analysis and modeling in machine learning and data analysis tasks.	Real-time work in this field Real-time features cannot increase system precision.

III. THEORETICAL FRAMEWORK

3.1 Proposed Method

Twitter holds significant importance as a social network platform where diverse topics are discussed, often serving as a primary source of daily news for many individuals. However, spammers and spam messages hinder the user experience by inundating them with fake and undesirable content. Over time, several automated methods have been introduced to combat spam in social networks, specifically on Twitter, each with advantages and limitations.

In the proposed approach, the researchers aim to present a new methodology for detecting spam tweets by leveraging the content-based features of tweets, along with features derived from the communication graph and informative account details. The process initiates with pre-processing the tweet contents, followed by the crucial task of identifying and extracting pertinent features. Subsequently, the NB classifier is employed to classify the tweets into two distinct categories: spam and non-spam, as depicted in Figure 1.

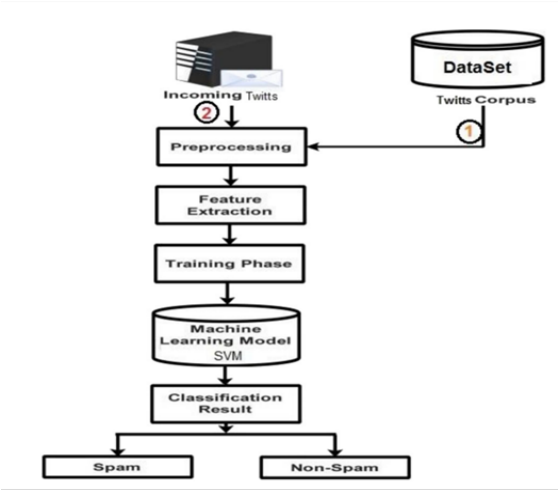


Fig.1. The structure of proposed method

In the practical realm, once the system receives the dataset and undergoes training using the NB algorithm, we obtain a trained system capable of differentiating between spam and non-spam data. Upon the arrival of a tweet into the system, the extracted features are utilized to determine its classification as either spam or non-spam. This system is hosted on a server, which, upon receiving a tweet, proceeds to evaluate whether it falls into the spam or non-spam category. If the tweet is classified as spam, it may be segregated into the spam section, similar to the current functionality observed in Gmail. Overall, depending on the classification outcome, the social network platform can delete the tweet, retain it, or display it within a specific section.

3.2 Tweets Pre-processing

During this phase, a series of operations are carried out on Tweets to prepare them for the subsequent feature extraction stage. Pre-processing, a crucial step in text processing, plays a significant role in determining the accuracy and effectiveness of subsequent processing steps.

3.3 Word Concurrence

Spammers frequently employ specific deceptive words to entice users, and being aware of these words can aid in spam detection. However, the detection of simultaneous events can be highly effective. In the suggested approach, we incorporate significant n-grams that spammers commonly utilize. N-grams refer to sequences of n words in the text, typically called 1-grams, 2-grams, etc. Identifying n-grams can significantly assist in identifying patterns of spammers' behavior. The detection of n-grams is accomplished using Latent Semantic Analysis (LSA).

3.4 Lexical Density and Semantic Diversity

In quantitative linguistic analysis, vocabulary richness refers to the extent of vocabulary utilized in a given text. It reflects the number of different words present in the text, with higher richness indicating a greater vocabulary diversity. Two evaluation metrics, namely Type Token Ratio (TTR) and Mean Word Frequency, are employed to assess the quality of vocabulary in both spam and non-spam tweets.

Through the examination of spam and non-spam samples within the dataset, it is observed that spammers tend to employ a limited vocabulary with reduced variety. Their focus on specific objectives, such as promoting products and increasing follower count, contributes to the repetitive nature of their language usage. Conversely, non-spam users are expected to exhibit a broader range of vocabulary due to the diverse topics they discuss.

TTR serves as a criterion for determining vocabulary richness in a text, distinguishing between the usage of varied words in spam and non-spam messages.

For dataset D, the TTR criterion is calculated by Eq. (1)

$$TTR = \frac{unique\ token\ in\ D}{tokens\ in\ D} \quad (1)$$

The lexical density criterion (LD) is also obtained by Eq. (2).

$$LD = \frac{words\ in\ D\ excluding\ stopwords}{tokens\ in\ D} \quad (2)$$

Equation (2) indicates that Lexical Diversity (LD) calculation excludes stop words. Stop words are commonly used words that are frequently repeated and do not carry significant semantic meaning in the text.

3.5 Stop Words

Stop words are a set of words that are typically removed during natural language data processing. These words, commonly called "stop words," are the most frequently used words in a language. It is important to note that there is no universally accepted list of stop words employed by all-natural language processing tools, and different tools may have their variations or may not utilize them at all.

3.6 User Reference

The user mentioning or tagging process involves a user directly addressing and including the user ID of another individual in a tweet. However, it should be noted that indiscriminate user mentioning is a strategy often employed by spammers to expand their reach and increase their follower count. In contrast, upon analyzing available datasets, it becomes evident that spammers tend to mention a much broader range of users without a specific relationship. In contrast, non-spam users typically mention a specific domain of individuals with whom

they share a specific connection. Furthermore, distinguishing between spammers and legitimate users can be achieved by examining the usernames and displayed names of users, as spammers often employ numeric characters in these fields.

3.7 Feature Selection

Tweets have a limited lifespan and remain visible briefly, typically around seven days from publication. In the context of spam detection, previous works have often focused on utilizing features derived from the historical data of tweets. However, considering the transient nature of tweets, these historical features may need more utility. Instead, incorporating dynamic and real-time features can significantly enhance the precision of the spam detection system. To this end, the applied features undergo the following classification process:

Table.1. A list of English stop words [16]

A	**Against**	**Am**	**Besides**	**Couldn't**	**Down**
Abaft	Agin	Amid	Best	Couldst	During
Aboard	Ago	Amidst	Better	D	Durst
About	Aint	Among	Between	Dare	E
Above	Albeit	Amongst	Betwixt	Dared	Each
Across	All	An	Beyond	Daren't	Early
Afore	Almost	And	Both	Dares	Either
Aforesaid	Alone	Anent	But	Daring	Em
After	Along	Another	By	Despite	English

The features mentioned earlier can be categorized into two classes: primary features and derivative features. Basic features are directly extracted without undergoing any additional processing. On the other hand, derivative features are derived by combining other features and performing computations. Examples of derivative features include sentiment analysis-based features and entropy calculations. Furthermore, these features can be further classified as static or dynamic. Static features remain unchanged after the creation of an account, whereas dynamic features have the potential to change over time. For instance, the user ID is static, while the user status is dynamic. A comprehensive list of these features is provided in Table 2.

ID	**Feature name**	**Status**	**Description/Definition**
F1	Account Age	static	Account lifetime
F2	Followers Count	dynamic	Number of followers
F3	Friends Count	dynamic	Number of friends
F4	Statuses Count	dynamic	Number of states

3.8 User Related Features

User Profile Features encompass details about the username, the displayed name, location, and other relevant information about the user's identity and characteristics. Account Information Features include attributes such as the date and time of account creation and indicators such as the presence or absence of a verification flag, among other pertinent details.

User Activity Based Features comprise a collection of attributes that provide insights into the user's behavior and engagement on the Twitter platform. These encompass metrics such as the number of friends or followers, the number of statuses or tweets posted, the nature or content type of the tweets, the timestamps indicating the creation time of tweets, and various other relevant factors that reflect the user's activity and interaction patterns.

Features based on the activities of others encompass attributes that go beyond the user's actions and are influenced by the activities of other users. These features consider factors such as the extent of viewership received by a person's tweet, the ability of a user to attract followers based on their friends' activities, and various other metrics that reflect the influence and engagement of the user within their social network. Examples of these features include the number of followers, the number of followings, the number of shared interests, the number of retweets, and other relevant indicators.

3.9 User Behavior Features

Examining their working behavior to distinguish between spammers and non-spammers based on their activity patterns is crucial. Spammers engage in many activities quickly, whereas non-spammers gradually increase their activity levels after creating an account. Additionally, the verification status of a user account can serve as a valuable feature in spam detection. Accounts that Twitter has verified are doubtful to be associated with spamming activities. This verification feature can be employed during the training process to identify non-spammers, and it can also be determined by experts when constructing a dataset.

3.10 Semantic Features

Content-based features are derived from the contextual information embedded within tweets. These features have been extensively utilized in numerous prior studies. In the present method, we incorporate two crucial features introduced as follows.

3.11 Accurate Response Rate

Because most typical accounts connect with their friends most of the time, the reply rate is used as an effective characteristic in the proposed strategy. On the other hand, Spammers typically transmit URL links and have shorter, more one-sided conversations. To determine whether an account has replied to friends or other users, use the $R_{\text{Reply Correct}}$ feature. The user will receive a suitable response if they are on their friend's list. The excellent response rate is calculated by Eq (Al-Zoubi et al., 2018) [3]:

$$R_{Reply-Correct} = \frac{N_{reply-Correct}}{T} \quad (3)$$

where T is the total number of tweets sent by that person, and $N_{\text{reply-Correct}}$ is the number of correct replies from that user. This figure is significantly lower for spammers than it is for regular users.

3.12 Rate of Variation

Spammers generally utilize a customized Twitter API application to send more messages and control their spam accounts. Spammers only utilize a few specific APIs; hence this rate is lower than regular users, even though normal users also use many applications based on their needs. Consequently, $R_{\text{API-Variety}}$ calculates the proportion of applied APIs to all APIs (Sedhai et al., 2018).

3.13 Features Based on Communication

Communication-based features are typically used to identify spammers who are attempting to avoid being detected by profile-based features. Computing the similarity of users with their neighbors using the SRank algorithm is one of the unique features that have yet to be used. This feature is essential for identifying spammers since social network analysis uses a hierarchical approach to determining the likelihood of each user interacting with other users. The current network can be viewed as a graph, G = (V,E), if each Twitter account is a node and every friendship connection is an edge. Spammers cannot change their position in the graph even if they can alter their tweeting or following habits.

To obtain similarity, first, the availability value is defined. Let Pp be N×N probability matrix for graph G. Matrix length is p. The availability from a to b is defined by Eq. (4).

$$H(a,b) = w_1 \times P^1_{a,b} + \ldots + w_p \times P^p_{a,b} + \ldots + w_{n-2} \times P^{n-2}_{a,b} \quad (4)$$

The weight of all pathways with length i is called w_i. $P^p_{a,b}$ is the probability of traveling from point a to point b along pathways of p-length, as determined by Eq. (5):

$$P^p_{a,b} = \frac{k_p(a,b)}{\sum_{\forall x \in G-\{a\}} k_p(a,x)} \quad (5)$$

where $k_p(a, b)$ is the quantity of p-length pathways connecting a and b. Since it takes a while to find every path in the graph with a different length, H(a,b) is changed to $H_s(a,b)$, which is then produced by Eq. (6):

$$H(a,b) = w_1 \times P^1_{a,b} + \ldots + w_s \times P^s_{a,b} \quad (6)$$

To achieve accurate results, the weight assigned to the short paths must be larger than the weight given to the long paths. These weights are obtained by Eq. (7).

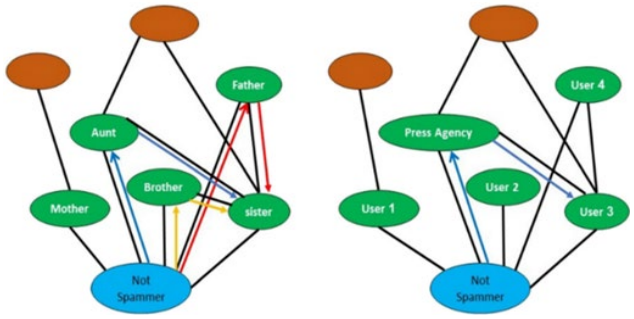


Fig.2. The number of two-length pathways for the spammer and the average user in the graph

$$w_p = 2^{s-p} \quad (7)$$

$H_s(a,b)$ is normalized through the lowest (H_{Min}) and maximum (H_{Max}) similarity in order to measure the similarity between two nodes, a and b, in the given graph more straightforward to understand. The SRANK-based similarity value is determined by Eq. (8). Here, the short path length is represented by the index s.

$$Srank_s(a,b) = \frac{H_s(a,b) - H_{Min}}{H_{Max} - H_{Min}} \quad (8)$$

In the suggested method, the given graph is formed after choosing a user's friends as neighbors to obtain SRANK. The number of pathways with lengths 1 and 2 per neighbor is the final metric to determine how similar each account pair is to the others. When comparing a user's mean similarities across all their neighbors, spammers have considerably lower mean similarities than regular users.

A- Common neighborhood rate

Another graph-based feature developed in this study, the ordinary neighborhood rate, shows how many friends a user's neighbors have in common. The relationships between users in the usual range are continuous. In turn, the social network of the spammers' surrounding nodes is treated as a different cluster because they do not know each other. The Rcn rate for spammers is considerably lower than those of regular users, making this a valuable feature to detect them, as determined by the standard neighborhood rate Eq. (9)

$$R_{cn} = \frac{\sum_{cn} v_n}{k_v \times \sum_n k_{v_n}} \quad (9)$$

where $\sum_{cnvn}$ is the number of familiar neighbors of neighbors v. k_v is the sum of the neighbors of the vertex v, and $\sum_{nkvn}$ is the sum of the neighbors related to each neighbor of vertex v.

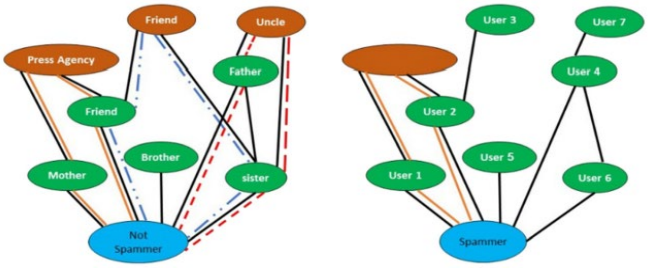


Fig.3. Differences between spammers and regular users in terms of the number and type of neighbors

As depicted in Fig. 3, a typical user has more friends in common with their neighbors. As a result, standard accounts—which frequently have quadrilateral shapes—have more significant social ties than spammer accounts.

3.14 Categorization of Tweets

The last action is carried out in this section. Numerous text pre-processing techniques have been applied, and various features have been retrieved for each tweet. The expert divides tweets into classifications for spam and non-spam. Using support vector machine classifiers, we aim to create a machine learning system that can identify correlations between features and spam and non-spam tweets. The machine learning algorithm carries out the learning process. After learning, the algorithm must accurately determine whether a new tweet with associated features is spam.

IV. Results And Discussion

4.1 Dataset

The suggested approach is implemented and evaluated using the Honeypot dataset [14]. This dataset contains information on users and their tweets gathered from Twitter. Over nine months, data on 22,223 spammers and 19,279 actual users were gathered. 2,353,473 spam tweets and 3,259,693 non-spam tweets are included in this data. Various times have been used to record the number of followers. As a result, the performance of the suggested strategy may be precisely determined using these data. These six text files comprise this dataset.

- This dataset contains information on users and their tweets gathered from Twitter. Data on 22,223 spammers and 19,279 actual users were gathered for nine months. 2,353,473 spam tweets and 3,259,693 non-spam tweets are included in this data. Various times have been used to record the number of followers. As a result, the performance of the suggested strategy may be precisely determined using these data. These six text files comprise this dataset.

- Content polluters Followings.txt displays each spammer's followers.

- Tweeters, tweet number, tweet text, and tweet creation date are all included in the content polluters tweets.txt file.

Table.3. A portion of the dataset's spam tweets file's content

Spammer ID	Tweet ID	Tweet Content	Publish Date
8905	6,248,723,047	COVID UPDATE: Looking for a range of solutions for our year-end...	10-11-2020 15:14:31
4263	8,532,862,174	PANDEMIC OVER: The China has created a vaccine…	21-04-2021 03:33:54

Table 3 displays several spam tweets associated with the tweets.txt file from content polluting.

4.2 Application of the Suggested Approach

Pre-processing is crucial for text processing projects, as discussed in section 3, mainly if the text contains distinct properties and a distinctive structure. According to their characteristics, tweets also make automated processing more difficult. Python and Java are used to accomplish the suggested method. The Java-based pre-processing procedure includes removing the less critical stop words. Python is used to implement the remaining process.

4.3 Setup of Parameters

In a supervised learning process, the classifier looks for a latent relationship between the characteristics and the target class using a variety of data that have been expertly labeled. The features of each tweet and tweeter are extracted for the used dataset after pre-processing. Based on prior experience, the

expert judges which tweets are spam and which are not. The support vector machine is now selected to use this labeled data as input. Model parameters and super-parameters are the two parameters used in the learning process. Model parameters are measured during the learning process and are ultimately classified by the classifier. However, it is the designer who must initially choose the superparameters. To provide the classifier with the best efficiency, these parameters are frequently chosen through trial and error.

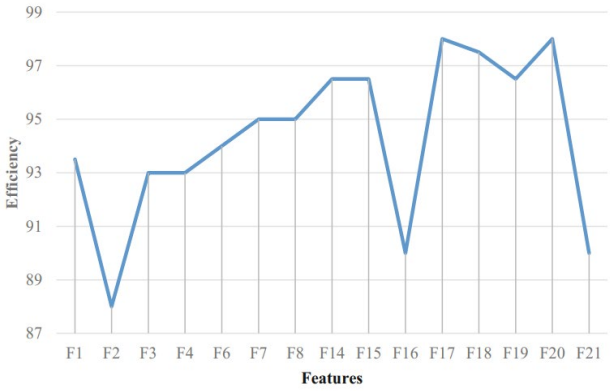
4.4 Cross-Checking

Overfitting is a concern that could arise in any learning process. Overfitting occurs when the learning process depends too much on the training data. On training data, it is very accurate, but on test data, it needs to be more accurate. Cross-validation employs a variety of strategies to avoid this. The training data are divided into k classes for cross-validation. The training procedure is carried out k times, and in each instance, the k-1 class is regarded as training while a different class is regarded as testing. More data is used to accomplish this learning process, and the overfitting issue is largely resolved. In the suggested approach, k is taken to be 10. K is a super-parameter based on the classification of parameters.

4.5 Features Choice

The choice of a subset of outstanding features is one of the critical steps in the classification process. Finding the best subset of characteristics when an issue has many features is extremely difficult. Section 3 presents several features that have been implemented, including details on tweets and tweeters. There are various ways to determine which properties are crucial to a classifier. The use of recursive feature reduction is one of the easiest approaches. The value of various attributes is determined using the same methodology. All features are initially considered using the recursive feature elimination approach before categorization is done. Then, the features are eliminated one by one, and the training is carried out without the removed feature. The classifier's efficiency is then determined. If there is little change in efficiency, it indicates that the elimination method is not more effective for the classifier and the feature can be removed from the final list of features; however, if eliminating a feature causes the classifier's efficiency to drop significantly, it indicates that the particular feature is crucial. The outcome of using the recursive feature reduction strategy for several features is shown in Figure 4. The depicted features are considered based on the criteria stated in Table 2. The F2, F16, and F21 features, or FollowersCount, LexRichWithUU, and LexRichWithoutUU, respectively, have the highest values for data categorization, according to the results. An ideal subset of features can then be chosen as a result. Finally, in the suggested strategy, the support vector machine classifier chooses 15 of the best characteristics to use in its learning process. Features F1, F3–F12, F19, F20, F23, and F24 are the best ones.

Fig.4. Recursive feature elimination is used to compare the values of features



4.6 Missing Values

There are empty values in the Honeypot dataset. In the data cleansing process, missing values and the issues they cause are particularly prevalent. Many approaches have been put out to deal with missing data in datasets and prevent issues brought on by it. The most popular and straightforward approach to this issue is to disregard any case lacking data for any attributes needing assessment. We will then have a dataset without missing values, which we may process using accepted techniques. However, this approach has a significant flaw in that, occasionally, eliminating missing values may result in ignoring a sizable portion of the original sample.

We outlined the specific mechanism that was employed. The mean approach is utilized in our paper to process The Honeypot dataset has missing data.

4.7 Evaluation Standards

To assess the effectiveness of the suggested method, the traditional performance metrics of precision, recall, accuracy, and f-measure are used. These characteristics are evaluated

using four well-known metrics listed in Table 4. In light of Table 4, we develop these standards [15].

The number of actual spam tweet detections is shown in Table 4 as TP. The number of non-spam cells mistakenly presented as spam have been counted as having FP—the number of non-spam detections that are reported as non-spam is known as the TN. The number of spam tweets mistakenly labeled as non-spam is known as FN. The primary comparison criteria are established per these factors by the Eq. 10 to 13 (Liu et a., 2019).

$$precision = \frac{TP}{TP + FP} \quad (10)$$

$$f\text{-}measure = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} = \frac{2 \times precision \times recall}{precision + recall}$$

$$recall = \frac{TP}{TP + FN} \quad (11)$$

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (12)$$

The system recall reflects the ratio of accurate spam tweet detections to all other tweet detections. In contrast, the system precision displays the ratio of genuine spam tweet detections to all other tweet detections. For both spam and non-spam tweets, the accuracy is the proportion of actual detections to all detections. Since increasing one decreases the other due to the negative relationship between precision and recall, we define the F-measure as Eq [16]. The geometric mean of the two criteria makes up this criterion.

Table.4. Abbreviations explained in evaluation

False	True	
FP	TP	Detect as spam
FN	TN	Detect as non-spam

4.8 ROC curve and the AUC standard

Area under curve (AUC) is a crucial metric to assess a classifier's effectiveness. The AUC stands for the receiver operating characteristic (ROC) area under the ROC diagram, where the higher the value of a classifier, the more influential the resulting classifier is. The effectiveness of the classifiers can be evaluated using the ROC diagram. also have a maximum value of one

Unlike other metrics for measuring classifier effectiveness, the AUC is independent of the classifier's decision threshold. As a result, this criterion, which cannot be computed with other performance evaluation criteria for classifiers, represents the dependability of an output of a specific classifier for various datasets. In situations where the classifiers' values are not comparable, it is not preferable to apply the AUC criterion. In some instances, the area under the ROC curves of these two classifiers are the same, but their value is different for different applications. Due to this, it does not appear logical to employ a metric or criterion other than the cost matrix. Last but not least, it is essential to remember that in addition to the criteria that were all calculated, the classifier's precision, the final complexity, and

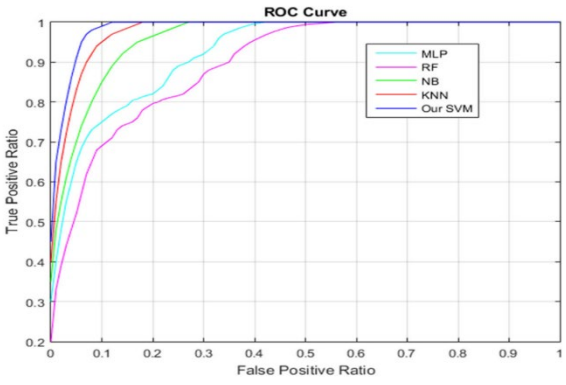


Fig.5. The area underlying the ROC diagram.

The interpretability of the learned model is crucial in interpretable classifiers like rule-based or decision-tree-based classifiers.

4.9 Applying a classifier using a Naïve Bayes

The primary goal of the suggested strategy is to extract the features correctly. We have attempted to extract several different attributes to characterize the tweets better. We achieve good extraction performance using the basic features and those retrieved from other features. The following stage involves feeding the Naive Bayes classifier the existing dataset with the retrieved features. Python also handles this stage of implementation.

However, extending to multi-class problems is simple, transferring the data to a new space using various kernel functions and categorizing various forms of data in any situation. This implementation makes use of kernel functions that are both polynomial and Gaussian. The polynomial kernel and the Gaussian RBF are often used NB kernels in Eq (14) and Eq (15), respectively. The polynomial kernel is utilized for the current data classifier since it demonstrated the highest performance for the existing dataset according to the tests and the results extracted from them.

$$K(x,y) = \left(xT_y + 1\right)^p \tag{14}$$

$$K(x,y) = exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right) \tag{15}$$

4.10 Evaluation Conclusions

After implementing the suggested approach, we assess the outcomes and compare them to other methods.

4.10.1 Setup of the System Hardware

An Intel Core i5 processor, Nvidia Geforce graphics card, and 8G RAM PC running Windows 10 64-bit are utilized to implement the suggested method. Table 5 displays the hardware configuration of the system in use.

4.10.2 Kernel Option

The data are randomly divided into 10 folds for the learning process, and the classifier then applies 10-fold cross-validation to the learning process. The support vector machine serves as a classifier in the suggested approach. In the first test, the learning process is carried out twice, once by the polynomial kernel and once by the Gaussian kernel, to assess the effectiveness of the two kernels. Table 6 presents the outcomes in terms of the evaluation criteria.

4.10.3 Comparison of the suggested approach with the k-nearest neighbors, multi-layer perceptron, support vector machine, and random forest techniques

The suggested method's outcomes are compared with those of many well-known classifiers for a more thorough assessment. Four classifiers—KNN, MLP, SVM, and RF—are considered in this. These classifiers also carry out the learning process. Table 7 displays the outcomes that were attained.

Table.5. Hardware configuration of the used system

Processor	Core i5
Memory	8 GB
Graphic	Nvidia Geforce
System	64 Bit

Table.6. Results of Gaussian and polynomial kernel evaluation

Method	**Precision**	**Recall**	**Accuracy**	**F-measure**
NB with Gaussian kernel	0.975	0.927	0.94	0.946
NB with polynomial kernel	0.988	0.953	0.96	0.969

Table.7. Evaluation findings for categorization using several techniques

Method	**Precision**	**Recall**	**Accuracy**	**F-measure**
MLP	0.952	0.926	0.93	0.933
KNN	0.984	0.948	0.963	0.962
SVM	0.965	0.96	0.957	0.96
RF	0.956	0.938	0.941	0.941
NB with polynomial kernel (proposed method)	0.988	0.953	0.96	0.969

The proposed technique has the highest precision and F-measure, according to the evaluation findings of the proposed method and other comparable methods (Table 7). Additionally, it is close to the maximum values for recall and accuracy. Regarding recall, the SVM technique performs best, and in terms of accuracy, the KNN method outperforms the proposed method by 0.003. In general, the NB classifier may be more effective for two-class issues if its features are comprehensive.

4.10.4 Comparison of the suggested method's ROC diagram with existing techniques

In Fig. 5, the proposed method and other methods are displayed on the ROC diagram. As observed, the proposed diagram is in the lead and has a higher AUC. The KNN approach is the one that is closest to the suggested method, among other methods.

As a result, the suggested strategy performs better than others. Table 8 displays the specifics of the AUC, also known as the area under the ROC diagram. As can be shown, the proposed technique has a higher AUC criterion than the competing methods.

Table.8. Comparison of the ROC diagram's area under the curve

Method	**AUC**
MLP	0.926
KNN	0.981
SVM	0.972
RF	0.938
MLP [13]	0.895
NB [13]	0.863
NB+MLP [13] Features	0.876
NB with polynomial kernel (proposed method)	0.985

V. Conclusions And Recommendations

In general, automatic text processing, especially spam detection in social networks, is a subject of enormous importance, and there are still many unresolved problems. We can split the task into two pieces to expand our suggested strategy in the future. Semantic and ontological features can augment the semantic meaning of features before extracting them. Second, NB can be used with other evolutionary computing techniques, such as genetic or SVM algorithms, to improve the classifier's performance. Apply additional improvements to the model by investigating preprocessing methods and machine learning

training adjustments that might boost the model's accuracy even with these improvements. Check for any effects on the model and look into early stopping. Apply the method to time-based systems and issues, such as simulations, email, SMS, and other spam detection systems.

REFERENCES

[1]. Alsaleh M, Alarifi A, Al-Quayed F, Al-Salman AS (2015) Combating comment spam with machine learning approaches. International Conference on Machine Learning and Applications (ICMLA):295–300.

[2]. Heydari A, Tavakoli MA, Salim N, Heydari Z (2015) Detection of review spam: A survey. Expert Syst Appl. 42(7):3634–3642.

[3]. R.G. Jimoh, A.M. Oyelakin, I.S.Olatinwo, K.Y. Obiwusi, S. Muhammad-thani, T.S. Ogundele, A. Giwa-raheem, O.F. Ayepeku, "Experimental Evaluation of Ensemble Learning-Based Models for Twitter Spam Classification" 5th volume, 2022.

[4]. Heru Agus Santoso, Eko Hari Rachmawanto, Ulfa Hidayati, "Fake Twitter Account Classification of Fake News Spreading Using Naïve Bayes" 7th Volume, pp. 228 - 23, 2020.

[5]. Ainul Yaqin, Beta Priyoko, "Implementation of Naive Bayes Algorithm for Spam Comments Classification on Instagram" pp. 508 - 513 2019.

[6]. Danezis G, Mittal P (2019) SybilInfer: detecting sybil nodes using social networks. Network and Distributed System Security Symposium (NDSS). In: San Diego. USA, California.

[7]. Gao H, Hu J, Wilson C, Li Z, Chen Y, Zhao BY (2010) Detecting and characterizing social spam campaigns. 10th ACM SIGCOMM Internet Measurement Conference (IMC), Melbourne, Australia:35–47.

[8]. Benevenuto F, Magno G, Rodrigues T, Almeida V (2010) Detecting spammers on Twitter. Collaboration. Electronic messaging, Anti-Abuse and Spam Conference (CEAS), p 6.

[9]. Varol O, Ferrara E, Davis CA, Menczer F, Flammini A (2017) Online human-bot interactions: detection,estimation, and characterization. International AAAI Conference on Web and social media, AAAI Press, pp. 280–289.

[10]. Chen C, Wang Y, Zhang J, Xiang Y, Zhou W, Min G (2017) Statistical features-based real-time detection of drifted Twitter spam. IEEE Transactions on Information Forensics and Security 12(4):914–925

[11]. Liu S, Wang Y, Zhang J, Chen C, Xiang Y (2017) Addressing the class imbalance problem in Twitter spam detection using ensemble learning. Comput. Secur. 69:35–49.

[12]. McCord M, Chuah M (2019) Spam detection on Twitter using traditional classifiers. In: Calero JMA, Yang LT, Mármol FG, García Villalba LJ, Li AX, Wang Y (eds) Autonomic and Trusted Computing (ATC), vol 6906, pp 175–186.

[13]. Subba Reddy K, Srinivasa Reddy E (2019) Detecting spam messages in Twitter data by machine learning algorithms using cross validation. International Journal of Innovative Technology and Exploring Engineering (IJITEE).

[14]. Lee K, Eoff BD, Caverlee J (2011) Seven months with the devils: a long-term study of content polluters on Twitter. Fifth International Conference on Weblogs and social media:185–192.

[15]. Wu T, Liu S, Zhang J, Xiang Y (2017) Twitter spam detection based on deep learning. Australasian Computer Science Week Multiconference, (3):1–8.

[16]. Inuwa-Dutse I, Liptrott M, Korkontzelos I (2018) Detection of spam-posting accounts on Twitter. Neurocomputing 315:496–511.