Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Comparing the Effectiveness and Efficiency of Machine Learning Models for Spam Detection on Twitter

Stephanie Chua[1*], Amy Tan[1], Puteri Nor Ellyza Nohuddin[2], Mohd Hanafi Ahmad Hijazi[3]

[1] Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia
[2] Higher Colleges of Technology, Sharjah Women's College, 79799 Abu Dhabi, United Arab Emirates
[3] Faculty Of Computing and Informatics, Universiti Malaysia Sabah, 88400 Kota Kinabalu, Sabah, Malaysia

ABSTRACT

A comprehensive study focused on the efficiency and effectiveness of machine learning models for Twitter spam detection was presented in this research. Spam detection on social media platforms is not only vital for user experience but also poses computational challenges due to the vast and dynamic nature of Twitter data. This investigation encompassed a range of machine learning models, including Naive Bayes (NB), Support Vector Machine (SVM), Logistic Regression (LR), k-Nearest Neighbours (KNN), and Decision Trees (DT). Their performances were scrutinized across two critical dimensions: classification accuracy and computational efficiency, as measured by the time taken for model execution. The results of the analysis revealed valuable insights into model performance. The NB and LR models emerged as the most computationally efficient models, with execution times ranging from 1.016 to 1.949 seconds. These models offered an attractive balance between speed and accuracy, making them suitable for real-time or resource-constrained applications. SVM, LR, KNN and DT were effective in classification with a performance of 98%. However, SVM models demanded longer execution times, ranging from 7.670 to 37.657 seconds. KNN and DT stroked a balance between accuracy and efficiency, with execution times ranging from 2.852 to 10.941 seconds and 1.080 to 2.442 seconds, respectively. Our research underscores the importance of considering both model effectiveness and computational efficiency when selecting a Twitter spam detection model. By offering a comparative assessment of these models, this study equipped researchers with valuable insights for making informed decisions in Twitter spam detection. It highlighted the trade-offs between model performance and efficiency, paving the way for more effective and resource-conscious approaches to combating spam on social media platforms.

Keywords:
Twitter; spam; text mining; machine learning

1. Introduction

Spam can be defined as unsolicited bulk messages or messages delivered to a group of people who did not request them [1]. Spam is frequently delivered via email. However, as the popularity of the social media platform increases, now it can also be delivered through social media, text messages, and phone calls. With the advancement of the Internet, spam in social media has increased

[*] *Corresponding author.*
E-mail address: chlstephanie@unimas.my

https://doi.org/10.37934/araset.61.2.127138

significantly over the years. This was one of the issues highlighted in a review by Abkenar *et al.,* [2] One of the most widely used social media platforms, Twitter, is also facing the issue of spamming. With the addition of individuals and organizations using the social networking service Twitter, spam on Twitter has become a significant concern because it is projected to increase continuously. There are many different types of ways that spammers can act on Twitter. For instance, direct message malware, reply spam, get more followers spam, trending topics spam, follow the spam, business opportunity spam, hashtag spam, direct message advertisements, brand tweet spam, and so on [3]. All these caused the spam on Twitter to continue to grow. This had become the cause of nuisances for Twitter users. Thus, spam should be detected and stopped before it can propagate. An automated approach is needed to detect spam tweets due to the overwhelming number of tweets. As such, a machine learning approach can be employed to learn a model for automatically detecting spam tweets. Many notable works on using machine learning to detect spam on Twitter have been published [4-8].

This research focused on identifying word features for classifying spam and non-spam tweets and utilizing these features in a machine-learning approach to develop tweet classification models. A comparative analysis was conducted to assess the performance of these models in terms of effectiveness and efficiency. The paper is organized as follows: Section 2 provides an overview of previous work on spam detection on Twitter. Section 3 discusses the dataset and the methodology employed in this study. Section 4 presents the results and corresponding discussions. Finally, section 4 concludes the paper and outlines future research directions.

2. Related Works

Some related work was undertaken previously in spam detection on Twitter. Çıtlak *et al.,* [9] surveyed detecting spam accounts on Twitter. Their study examined prominent spam detection methods and compared and evaluated the process of distinguishing between real users and fake users, as well as the strengths and weaknesses of these methods.

Velammal and Aarthy [10] presented a system that was designed to identify spam tweets using four lightweight detectors: blacklist domain detector, near duplicate detector, reliable ham detector, and multiclass detector. Detected tweets underwent classification through ensemble classifiers, including naïve Bayes, logistic regression, and random forest. The final labels for classified tweets were determined using a voting method. The proposed system achieved a spam tweet detection accuracy of 79% using the naïve Bayes classifier, with the potential for further optimization as more sample data became available.

Rodrigues *et al.,* [11] proposed to develop a system that could determine whether a tweet was categorized as "spam" or "ham" and evaluate the emotional sentiment of the tweet. The extracted features, after preprocessing the tweets, were classified using various classifiers, including decision tree, logistic regression, multinomial naïve Bayes, support vector machine, random forest, and Bernoulli naïve Bayes, for spam detection. Stochastic gradient descent, support vector machine, logistic regression, random forest, naïve Bayes, and deep learning methods, including the simple recurrent neural network (RNN) model, long short-term memory (LSTM) model, bidirectional long short-term memory (BiLSTM) model, and 1D convolutional neural network (CNN) model, were employed for sentiment analysis. The performance of each classifier was analyzed. The classification results showed that the features extracted from the tweets could satisfactorily be used to identify whether a particular tweet was spam or not, and a learning model was created that associated tweets with specific emotional sentiments. The best results were demonstrated by the multinomial naïve Bayes classifier, that had achieved a classification accuracy of 97.78%, and the deep learning model, specifically LSTM, that had attained a validation accuracy of 98.74% for the Twitter spam classification.

Gupta *et al.,* [12] presented a system for tweet classification that incorporates user-based and tweet-based attributes, as well as tweet text. Four machine learning techniques were used, including Support Vector Machine, Neural Network, Random Forest, and Gradient Boosting. They collected 400,000 public tweets and classified 150,000 spam tweets and 250,000 non-spam tweets based on user-based features. The top 30 unique words were calculated for each word based on the collected information to determine a tweet's spam status. Thirteen lightweight features were extracted from the dataset, and multiple machine learning algorithms were used to train the model on this processed dataset. Three separate feature sets were used, and the classifiers' effectiveness was measured using Recall, Precision, F-measure, and Accuracy. Among all classifiers in feature set 1, the Neural Network achieved the highest accuracy, at 91.65%. Only the Support Vector Machine classifier was used in feature set 2 because it provided an input vector with dimensions of 100,000 features and was impractical for other classifiers. Random Forest outperformed the Neural Network by 2% in feature set 3, which was more user-based. Combining the top 30 unique words with user-based features allowed for spam tweet classification based on the tweet's text. Future work includes the use of a self-learning algorithm to update the Bag-of-Words model based on new spam tweets and the use of the URL crawling approach to detect Twitter spam. In real-time, Twitter spam could be identified by using the Frequent Pattern Mining of tweet text. These three techniques will be combined to address Spam Drift issues.

Udge *et al.,* [13] presented a spam detection classification model that combines binary classification and automatic learning techniques. To test their model, the authors used the Naïve Bayes classifier and Support Vector Machine classifier in experiments conducted on both online and offline spam datasets. In the offline dataset, the Naïve Bayes classifier achieved a higher accuracy of 94.40% compared to the Support Vector Machine classifier's 70.40% accuracy. In the online dataset, the Naïve Bayes classifier obtained an accuracy of 90.00%, while the Support Vector Machine classifier achieved 68.70% accuracy. These differences in performance were due to the imbalanced data bias in the datasets. To enhance spam detection rates, the authors recommend including additional discriminative features or using a better model in the system.

Santoshi *et al.,* [14] proposed the use of a Naïve Bayes classifier for spam detection. Their work discussed the analyzing and classifying of tweets as spam or ham (non-spam) based on the words used in the tweets. It aimed to focus on finding tweets that were considered spam. Various classification techniques were used to detect spam and fake tweets, such as Support Vector Machine (SVM), Extreme Learning Machine (ELM), K-Nearest Neighbors (KNN) and Fuzzy K-Means. However, the authors selected a Naïve Bayes classifier in their research as they believed that the algorithm can better separate ham and spam using posterior values. The process involved six steps to build a Naïve Bayes model, comprising libraries importing, pre-processing, feature extraction, feature matching, model construction and results evaluation. The Naïve Bayes classifier was able to achieve 95.70% of exactness and 95.70% in F-measure.

In their research, Biyani and Khan [15] examined several spamming attacks, strategies for detecting spam, campaigns on social networking sites, and information related to spam detection. The authors focused specifically on Twitter and presented an effective method for identifying and filtering out unwanted tweets. Their proposed system involved constructing a model based on probabilities to classify tweets as either "positive" or "negative." To conduct their experiment, they used the Social Honeypot dataset, which they obtained from Kaggle. This dataset contained 22,223 content polluters, their number of followers over time, and 2,353,473 tweets. Additionally, it included information about 19,276 legitimate individuals, their number of followers over time, and

3,259,693 tweets. The authors employed three different machine learning algorithms, namely Support Vector Machine (SVM), Decision Tree, and Logistic Regression, to compare their performances. According to the results of their experiment, the SVM classifier produced the highest accuracy (92.04%) for the parameter tweets in the dataset, outperforming the Decision Tree classifier (90.33%) and the Logistic Regression classifier (85.70%). In future work, the authors suggest that additional Twitter data could be collected using the Twitter API to provide more metadata and input features for machine learning.

Shen *et al.,* [16] introduced an innovative attention-based deep learning model for discerning social spammers on Twitter. Specifically, the state-of-the-art pretraining model, BERTweet, was introduced to acquire tweet representations. Subsequently, an original attention mechanism was proposed to capture user representations by distinguishing nuanced variations among tweets authored by each user. This model also integrated social interactions, employing a graph attention network to refine user representations, thereby enhancing the precision of spammer identification. Experimental assessments conducted on a publicly accessible real-world Twitter dataset validate the efficacy of the proposed model, demonstrating significant performance enhancement.

The main technique used by Oyelakin *et al.,* [17] was ensemble learning, specifically employing two categories of ensemble algorithms, to build Twitter spam classification models. The contribution was in investigating the behavior of machine learning-based models, constructed with both homogeneous and heterogeneous algorithms, for Twitter spam classification. The authors employed an ANOVA-F test to select promising features from the dataset. The primary contributions were the use of a homogeneous tree-based Random Forest (RF) ensemble and a heterogeneous ensemble vote classifier for Twitter spam classification. The homogeneous ensemble employed tree-based algorithms to construct a Twitter spam detection model, while the heterogeneous ensemble combined Support Vector Machine (SVM) and Decision Tree (DT) algorithms through a maximum voting classifier. The study concluded that the performance of the Twitter spam detection models was promising, with the heterogeneous model outperforming the homogeneous model in terms of accuracy, precision, recall, and F_1-score.

Bharati *et al.,* [18] introduced a spam classification model for Twitter using deep learning techniques that incorporated sentiments and topics. The process began with data collection, followed by preprocessing steps such as removing stop words, stemming, and tokenization. Feature extraction included capturing post tagging, headwords, rule-based lexicon, word length, and weighted holo-entropy features. The proposed approach involved extracting sentiment scores to analyze differences between spam and non-spam content. Subsequently, the classification of spam data on Twitter was performed using an Optimized Deep Ensemble method that integrated neural network (NN), support vector machine (SVM), random forest (RF), and convolutional neural network (CNN). The weights of the CNN were optimized using an arithmetic crossover-based cat swarm optimization (AC-CS) model. The performance of the developed approach was evaluated against existing methods. Notably, the proposed AC-CS+ ensemble model achieved higher accuracy, with an 18.1%, 14.89%, 11.7%, 12.77%, 10.64%, 6.38%, 6.38%, and 6.38% advantage over SVM, DNN, RNN, DBN, MFO+ ensemble model, WOA+ ensemble model, EHO+ ensemble model, and CSO+ ensemble model models, respectively, when the learning percentage was 80%.

Kaddoura *et al.,* [19] presented a comprehensive survey on the recent advancements in spam text detection and classification in social media. They discussed various techniques that were involved in spam detection and classification, including Machine Learning, Deep Learning, and text-based approaches. The challenges that had been encountered in identifying spam, along with its control mechanisms and the datasets used in previous studies on spam detection, were also presented. They highlighted the issues of the presence of sarcastic text, multilingual data, and improper labeling of the datasets.

In conclusion, the literature review encompassed a wide array of approaches and methodologies employed for Twitter spam detection. Various machine learning algorithms, including Naïve Bayes, support vector machines, decision trees, and deep learning models, were explored for this purpose. Notably, several studies emphasized the importance of feature selection and extraction to enhance classification accuracy. Additionally, ensemble learning techniques and innovative methods, such as attention-based deep learning models and sentiment analysis, were proposed to improve spam classification. Furthermore, efforts were made to address challenges like imbalanced datasets and the incorporation of user-based attributes. The findings from these studies revealed substantial progress in the field of Twitter spam detection, with classification accuracies ranging from 79% to 98.74%. While some models prioritized computational efficiency, others emphasized higher accuracy, demonstrating the importance of considering both aspects in real-world applications. Future research directions included the incorporation of self-learning algorithms, URL crawling approaches, and frequent pattern mining to address evolving spam detection challenges. Overall, this body of research provided valuable insights and methodologies for developing effective and efficient Twitter spam detection systems.

3. Methodology

3.1 Data Selection

During the data selection process, relevant data is collected to be used as the targeted input. For this project, the spam tweet dataset from Kaggle [20] was utilized. This dataset contained general spam found on Twitter. It comprised seven descriptive attributes and one class attribute: id, tweet, following, followers, action, is_retweet, and location, and a total of 11968 rows with eight columns of data. The class distribution was 6153 rows for non-spam and 5815 rows for spam. Table 1 shows the descriptions of the attributes in the dataset.

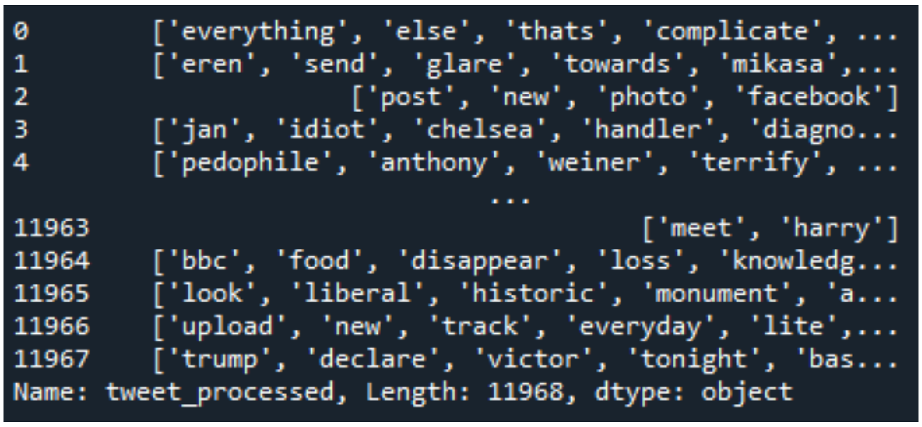
Table 1
Attribute description

Attributes	Description
Id	Identification that represents each tweet
Tweet	Tweet text
Following	The number of accounts that the tweeter is following
Followers	The number of accounts that follow the tweeter
Actions	The total number of favourites, replies and retweets of the said tweet
Is Retweet	Binary [0,1] value to indicate if the said tweet is a retweet
Location	Location provided by the user
Class	Non-Spam or Spam

3.2 Data Preprocessing

The tweets from the spam tweet dataset underwent text preprocessing to remove unnecessary data. The text preprocessing steps included Text Cleanup, Tokenization, Filtering, and Lemmatization. To begin, a function called "remove_punctuation" was applied to remove punctuation from the tweets, resulting in a new column called "tweet_remove_punctuation." The tweets were then subjected to Text Cleanup, which converted all uppercase tweets to lowercase and saved in the "tweet_lowercase" column. Next, Text Tokenization was performed on the tweets in the "tweet_lowercase" column, breaking down the tweet sentences into words by removing white spaces. The tokenized tweets were saved in the "tweet_tokenization" column. Following that, Text Filtering was applied to the tweets in the "tweet_tokenization" column to remove common stop words and symbols frequently occurring in the collected tweets.

The filtered tweets were saved in separate columns named "tweet_remove_stopwords" and "tweet_remove_symbols" respectively. Lemmatization was then applied to the tweets in the "tweet_remove_symbols" column to convert words to their base root forms. The lemmatized tweets were saved in the "tweet_lemmatization" column. Lastly, a regular expression, also known as "regex" or "regexp," was used to match specific text strings and replace them with blank spaces. In this case, the regular expression was used to search and substitute the occurrence of "amp" with a space to remove it. Once the text preprocessing was completed, all the processed tweets were saved in a new column named "tweet_processed." This column contained the cleaned tweets after the text preprocessing steps. Figure 1 shows the output of the “tweet_processed”.



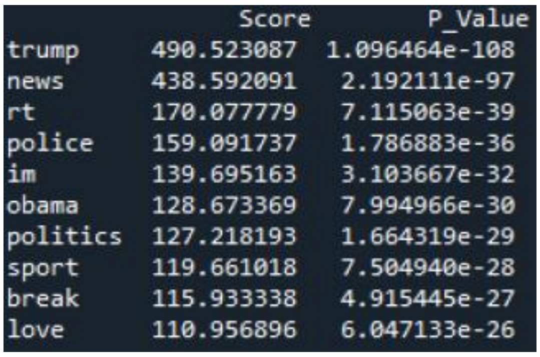
```
0        ['everything', 'else', 'thats', 'complicate', ...
1        ['eren', 'send', 'glare', 'towards', 'mikasa',...
2                     ['post', 'new', 'photo', 'facebook']
3        ['jan', 'idiot', 'chelsea', 'handler', 'diagno...
4        ['pedophile', 'anthony', 'weiner', 'terrify', ...
                                ...
11963                                  ['meet', 'harry']
11964    ['bbc', 'food', 'disappear', 'loss', 'knowledg...
11965    ['look', 'liberal', 'historic', 'monument', 'a...
11966    ['upload', 'new', 'track', 'everyday', 'lite',...
11967    ['trump', 'declare', 'victor', 'tonight', 'bas...
Name: tweet_processed, Length: 11968, dtype: object
```

Fig. 1. Output of “tweet_processed”

3.3 Data Transformation

In the data transformation stage, feature selection is applied to the preprocessed dataset. This step aims to create consistent data in a suitable format for the data mining phase, ultimately improving the accuracy and training time of the model. Before performing feature selection, the "tweet_processed" data was converted into a Document-Term Matrix (DTM) using the count vectorizer technique. A Document-Term Matrix is a mathematical representation that captures the frequency of terms present in a collection of documents. In this matrix, each row corresponds to a document, each column corresponds to a term (such as a word), and each value represents the occurrence of the term within the document. This transformation helps convert unstructured data into a structured format. In the context of the study, the frequencies of words in the "tweet_processed" dataset were transformed into binary values (0 or 1) to indicate their presence or absence.

Subsequently, feature selection using the chi-square test with SelectKBest was applied to the Document-Term Matrix. This technique selects the top 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100 features based on their scores and p-values. Figure 2 shows the top 10 features and their scores and p-values.

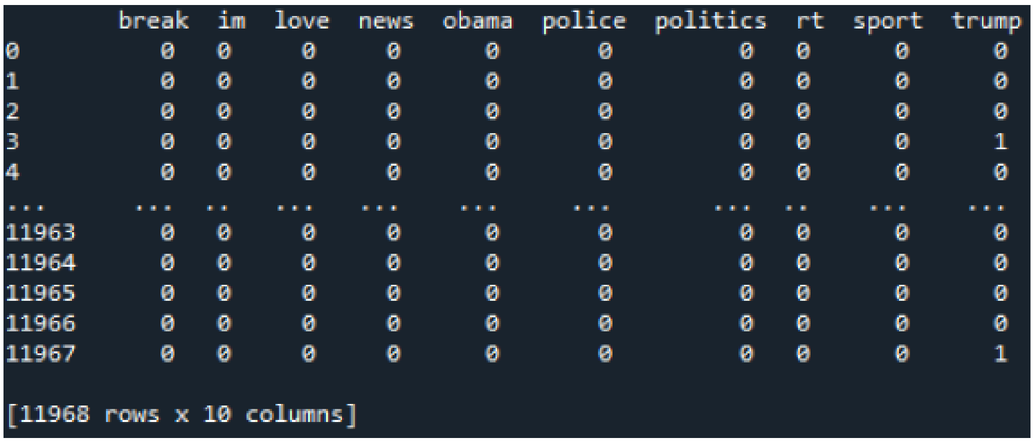


```
                Score        P_Value
trump      490.523087  1.096464e-108
news       438.592091   2.192111e-97
rt         170.077779   7.115063e-39
police     159.091737   1.786883e-36
im         139.695163   3.103667e-32
obama      128.673369   7.994966e-30
politics   127.218193   1.664319e-29
sport      119.661018   7.504940e-28
break      115.933338   4.915445e-27
love       110.956896   6.047133e-26
```

Fig. 2. Top 10 features and their scores and p-values

The selected features, along with their scores and p-values, were stored in a newly created data frame called "top10_features_df." Consequently, the dataset was reduced from 19,514 columns to 10 columns, retaining only the top 10 features based on the feature selection process. The original spam dataset was then merged with the "top10_features_df" Document-Term Matrix. Duplicate columns, namely "Tweet," "tweet_remove_punctuation," "tweet_lowercase," "tweet_tokenization," "tweet_remove_stopwords," "tweet_remove_symbols," "tweet_lemmatization," and "tweet_processed," were eliminated. Additionally, the "location" column was removed due to a significant number of missing values that couldn't be substituted with any other information. Lastly, the "Id" column was also excluded as it served as the identifier for each tweet.

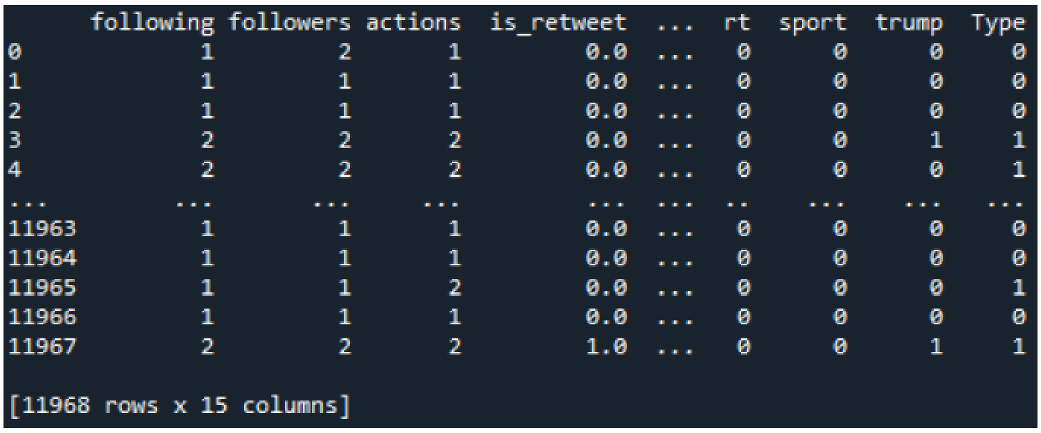
Data discretization was applied to the attributes "following," "followers," and "actions." Data discretization is a technique used to convert continuous attribute values into a finite set of ranges, with minimal loss of data. This process aids in enhancing data interpretation and makes data management more manageable. For the three attributes mentioned, a three-level discretization scheme was implemented. The levels were defined as "Low" (mapped to "1"), "Average" (mapped to "2"), and "High" (mapped to "3"). Each attribute was discretized by carefully considering its range and logically determining the appropriate discretized range. For instance, in the case of the "following" attribute, the range from "-1 to 160" was mapped to "Low," the range from "161 to 16,000" was mapped to "Average," and the range from "16,001 to 1,600,000" was mapped too "High". All the above processes were repeated for 20, 30, 40, 50, 60, 70, 80, 90, and 100 features. Figure 3 shows the Document Term Matrix for the top 10 features. The input data after data transformation is shown in Figure 4.



```
       break  im  love  news  obama  police  politics  rt  sport  trump
0          0   0     0     0      0       0         0   0      0      0
1          0   0     0     0      0       0         0   0      0      0
2          0   0     0     0      0       0         0   0      0      0
3          0   0     0     0      0       0         0   0      0      1
4          0   0     0     0      0       0         0   0      0      0
...      ...  ..   ...   ...    ...     ...       ...  ..    ...    ...
11963      0   0     0     0      0       0         0   0      0      0
11964      0   0     0     0      0       0         0   0      0      0
11965      0   0     0     0      0       0         0   0      0      0
11966      0   0     0     0      0       0         0   0      0      0
11967      0   0     0     0      0       0         0   0      0      1

[11968 rows x 10 columns]
```

Fig. 3. Document Term Matrix (DTM) for the top 10 features



```
       following  followers  actions  is_retweet  ...  rt  sport  trump  Type
0              1          2        1         0.0  ...   0      0      0     0
1              1          1        1         0.0  ...   0      0      0     0
2              1          1        1         0.0  ...   0      0      0     0
3              2          2        2         0.0  ...   0      0      1     1
4              2          2        2         0.0  ...   0      0      0     1
...          ...        ...      ...         ...  ...  ..    ...    ...   ...
11963          1          1        1         0.0  ...   0      0      0     0
11964          1          1        1         0.0  ...   0      0      0     0
11965          1          1        2         0.0  ...   0      0      0     1
11966          1          1        1         0.0  ...   0      0      0     0
11967          2          2        2         1.0  ...   0      0      1     1

[11968 rows x 15 columns]
```

Fig. 4. The input data after data transformation

3.4 Data Mining

After completing the pre-processing and transforming the dataset into a suitable format for data mining, the next stage involves implementing five different machine learning algorithms: Naïve Bayes classifier, support vector machine classifier, logistic regression classifier, K-nearest neighbor classifier, and decision tree classifier. The purpose of this stage was to analyze the models' ability to identify significant patterns in the spam dataset related to tweet classification.

Before applying the dataset to the machine learning algorithms, a splitting process was performed. The dataset was divided into two parts: X and Y. The X dataset comprised the selected features, including "following," "followers," "actions," and the top 10 features selected through chi-square feature selection. On the other hand, the Y dataset represented the class labels for tweet classification, which was the target variable. The dataset was then split into a 70% training set and a 30% test set. To maintain the class distribution, stratified splitting was applied, ensuring that the tweets were divided according to their classification class labels. This could enhance the accuracy of the classification. Additionally, a random state was set to ensure the reproducibility of results across program iterations. Once the training set and test set were prepared, five different machine learning algorithms were employed using the training set. K-fold cross-validation was used to evaluate the models' performances on multiple subsets of the training sets. In this case, a value of k = 10 was used for all five machine learning algorithms. Various parameters were also adjusted for the algorithms to achieve optimized performance. The parameter tuning for each machine-learning algorithm is described below:

i. Naïve Bayes (NB): In the NB algorithm, four NB techniques were compared; ComplementNB, GaussianNB, MultinomialNB and BernoulliNB. Preliminary tests showed that BernoulliNB had a higher accuracy compared to the others. Therefore, BernoulliNB was selected to be used in learning the model.
ii. Support Vector Machine (SVM): In SVM, four kernels were tested to get the best kernel for learning the model. The kernels used were linear, poly, rbf, and sigmoid. The rbf kernel showed the best performance in the preliminary tests.
iii. Logistic Regression (LR): In the LR algorithm, four solvers were applied to find the best parameters for its model. The four solvers used were lbfgs, liblinear, newton-cg, and saga. Among these solvers, liblinear solver performed the best with a maximum iteration (max_iter) value of 100 and a random state (random_state) value of 1.

iv. K-Nearest Neighbour (KNN): In the KNN algorithm, the number of neighbors, denoted as k, needs to be determined. There are two methods available to determine the value of k. The first method is to use the square root of the total amount of data in the training dataset (k = sqrt(N)), where N represents the total number of data points in the training dataset. It is important to select an appropriate value for k because using lower values may result in overfitting, while larger values may introduce higher processing complexity in distance calculations. For example, if N is equal to 8377 (the total number of data points in the training dataset), then k = sqrt (8377) ≈ 91.53. The second method involves utilizing GridSearchCV to determine the value of k. This method involves applying different dataset sizes to the model. By employing GridSearchCV, a distinct value of k will be determined for each dataset size. The model will be tested with different values of k, such as k = 91, k = 3, k = 5, k = 7, and k = 9, to identify the optimal value for the given dataset size. K=5 had shown to give the best results in the preliminary tests.

v. Decision Tree (DT): In the DT algorithm, two criteria (Gini and Entropy) and different maximum depths were employed to search for the most suitable parameters for the model. The Gini criterion measures the mislabeling of dataset features, while entropy quantifies the information disorder of features concerning the target variable. Since both criteria yield comparable results, it was difficult to determine which one performs better. To address this, pruning techniques were employed to identify the optimal maximum depth for the decision tree. The maximum depth varied depending on the dataset size being tested. Preliminary tests showed that gini criterion with the maximum depth of 15 and random_state of 1 performed the best.

4. Results

To assess the performance of the classification models, several evaluation measures were employed. The confusion matrix, a tabular representation, summarizes the classifier's correct and incorrect prediction results. It provided the computation of accuracy, precision, recall, and F_1-Score measures to assess the model's performance. The outcomes presented in this section were derived from utilizing the optimized parameters for each machine learning technique as discussed in the previous Sub-Section 3.4.

Table 2 shows the evaluation measures for each of the machine learning models, taking the minimum and maximum values across the results for the top 10 to 100 features used. The Naive Bayes model showed reasonably good performance, with accuracy, precision, recall, and F1-scores in the range of 77% to 84%. This range indicated some variability in performance, suggesting that the model's effectiveness might have depended on the number of features used to be able to capture the specific characteristics of the spam messages. While NB is known for its simplicity and speed, it may not have captured complex relationships in the data compared to more advanced models.

Table 2
Evaluation measures for the top 10 to 100 features for each model (min-max)

Model	Accuracy	Precision	Recall	F_1-Score
NB	77-84%	78-84%	77-84%	76-84%
SVM	98%	98%	98%	98%
LR	97-98%	97-98%	97-98%	97-98%
KNN	98%	98%	98%	98%
DT	98%	98%	98%	98%

The SVM model demonstrated outstanding performance with consistently high scores across all metrics. Its accuracy, precision, recall, and F1-score were all at 98%, indicating a robust ability to correctly classify spam tweets. However, it was crucial to consider the computational cost associated with SVM, especially when dealing with large datasets. The LR model exhibited strong performance, with accuracy ranging from 97% to 98%, precision ranging from 97% to 98%, recall ranging from 97% to 98%, and F1-Score ranging from 97% to 98%. The LR model also performed impressively, with accuracy, precision, recall, and F1-scores ranging between 97% and 98%. This model provided a strong balance between precision and recall, making it suitable for spam detection. LR offered computational efficiency, making it an attractive choice for large-scale Twitter data analysis.

KNN consistently achieved high scores across all metrics, with accuracy, precision, recall, and F_1-scores at 98%. KNN was known for its simplicity and ease of implementation, but it might have been sensitive to the choice of the number of neighbours (k) and could have become computationally expensive with large datasets. The DT model also demonstrated excellent performance, with accuracy, precision, recall, and F_1-Score of 98%. This performance matched the high scores of the SVM, LR, and KNN models. DT models were interpretable and could capture nonlinear relationships in the data, making them a valuable choice for this task.

Overall, the SVM, LR, KNN and DT models consistently outperformed the NB model in terms of accuracy, precision, recall, and F_1-Score. These models showed high accuracy and precision in classifying spam and non-spam tweets, and their ability to correctly identify the target class was reflected in their high recall values. Consequently, these models could be considered reliable options for the identification of spam and non-spam tweets.

Table 3 shows the time taken for classification for each machine learning model, taking the minimum and maximum values across the times for the top 10 to 100 features used. The NB model demonstrated the shortest time taken, ranging from 1.016 to 1.948 seconds. This indicated that the NB model was computationally efficient and requires minimal processing time for classifying the tweets. Despite its lower performance in terms of accuracy and other metrics, NB proved to be a quick and efficient option for tweet classification.

Table 3
Time taken for classification for each model (min-max)

Model	Time taken (s)
NB	1.016 - 1.948
SVM	7.670 - 37.657
LR	1.168 - 1.949
KNN	2.852 - 10.941
DT	1.080 - 2.442

The SVM model exhibited the longest time taken, ranging from 7.670 to 37.657 seconds. This suggests that the SVM model required more computational resources and processing time compared to the other models. However, the high accuracy and performance of the SVM model compensated for the longer time taken, making it a suitable choice when accuracy was the primary concern and time constraints were not critical. The LR model demonstrated a relatively short time taken, ranging from 1.168 to 1.949 seconds. LR showed a balance between computational efficiency and performance, making it a favorable option for tweet classification tasks that required both accuracy and reasonable processing time.

The KNN model exhibited a moderate time taken, ranging from 2.852 to 10.941 seconds. The KNN model struck a balance between computational complexity and performance, providing accurate classification results within a reasonable time frame. The DT model demonstrated a relatively short time taken, ranging from 1.080 to 2.442 seconds. Similar to the LR model, DT offered a favorable combination of accuracy and computational efficiency.

To summarize, the time taken by each model varies, with NB and LR being the quickest and SVM requiring the longest processing time. Depending on the specific requirements of the application, such as the need for fast predictions or a trade-off between accuracy and speed, the choice of model could be tailored accordingly.

5. Conclusions

Overall, considering the comparable performance of the SVM, LR, KNN, and DT models, along with their reasonable processing times, they proved to be promising choices for tweet classification tasks. The NB model, although slightly lower in performance, provided an efficient and time-effective option for situations where computational resources were limited or speed was a priority. The selection of the appropriate model would ultimately depend on the specific needs and trade-offs of the application at hand.

It is important to acknowledge the limitations of the research. Due to the dynamic nature of social media platforms like Twitter, the collected dataset may not capture all types of spam or be fully representative of all spam detection challenges. Future research can explore larger and more diverse datasets to improve the generalizability of the models. Additionally, the research can be extended to incorporate more advanced techniques, such as deep learning models and the integration of contextual information to further enhance spam detection accuracy.

Acknowledgement
The authors would like to acknowledge the Faculty of Computer Science and Information Technology (FCSIT), Universiti Malaysia Sarawak (UNIMAS) for the research opportunity and financial assistance.

References
[1] Abkenar, Sepideh Bazzaz, Mostafa Haghi Kashani, Mohammad Akbari, and Ebrahim Mahdipour. "Learning textual features for Twitter spam detection: A systematic literature review." *Expert Systems with Applications* 228 (2023): 120366. https://doi.org/10.1016/j.eswa.2023.120366

[2] Abkenar, Sepideh Bazzaz, Mostafa Haghi Kashani, Ebrahim Mahdipour, and Seyed Mahdi Jameii. "Big data analytics meets social media: A systematic review of techniques, open issues, and future directions." *Telematics and informatics* 57 (2021): 101517. https://doi.org/10.1016/j.tele.2020.101517

[3] Mukunthan, B., and M. Arunkrishna. "Spam detection and spammer behaviour analysis in Twitter using content based filtering approach." In *Journal of Physics: Conference Series*, vol. 1817, no. 1, p. 012014. IOP Publishing, 2021. https://doi.org/10.1088/1742-6596/1817/1/012014

[4] Madisetty, Sreekanth, and Maunendra Sankar Desarkar. "A neural network-based ensemble approach for spam detection in Twitter." *IEEE Transactions on Computational Social Systems* 5, no. 4 (2018): 973-984. https://doi.org/10.1109/TCSS.2018.2878852

[5] Tajalizadeh, Hadi, and Reza Boostani. "A novel stream clustering framework for spam detection in Twitter." *IEEE Transactions on Computational Social Systems* 6, no. 3 (2019): 525-534. https://doi.org/10.1109/TCSS.2019.2910818

[6] Senthil Murugan, N., and G. Usha Devi. "Detecting streaming of Twitter spam using hybrid method." *Wireless Personal Communications* 103, no. 2 (2018): 1353-1374. https://doi.org/10.1007/s11277-018-5513-z

[7] Concone, Federico, Giuseppe Lo Re, Marco Morana, and Claudio Ruocco. "Assisted labeling for spam account detection on twitter." In *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, pp. 359-366. IEEE, 2019. https://doi.org/10.1109/SMARTCOMP.2019.00073

[8] Adewole, Kayode Sakariyah, Tao Han, Wanqing Wu, Houbing Song, and Arun Kumar Sangaiah. "Twitter spam account detection based on clustering and classification methods." *The Journal of Supercomputing* 76 (2020): 4802-4837. https://doi.org/10.1007/s11227-018-2641-x

[9] Çıtlak, Oğuzhan, Murat Dörterler, and İbrahim Alper Doğru. "A survey on detecting spam accounts on Twitter network." *Social Network Analysis and Mining* 9 (2019): 1-13. https://doi.org/10.1007/s13278-019-0582-x

[10] Velammal, B. L., and N. Aarthy. "Improvised Spam Detection in Twitter Data Using Lightweight Detectors and Classifiers." *International Journal of Web-Based Learning and Teaching Technologies (IJWLTT)* 16, no. 4 (2021): 12-32. https://doi.org/10.4018/IJWLTT.20210701.oa2

[11] Rodrigues, Anisha P., Roshan Fernandes, Adarsh Shetty, Atul K, Kuruva Lakshmanna, and R. Mahammad Shafi. "[Retracted] Real-time twitter spam detection and sentiment analysis using machine learning and deep learning techniques." *Computational Intelligence and Neuroscience* 2022, no. 1 (2022): 5211949. https://doi.org/10.1155/2022/5211949

[12] Gupta, Himank, Mohd Saalim Jamal, Sreekanth Madisetty, and Maunendra Sankar Desarkar. "A framework for real-time spam detection in Twitter." In *2018 10th International Conference on Communication Systems & Networks (COMSNETS)*, pp. 380-383. IEEE, 2018. https://doi.org/10.1109/COMSNETS.2018.8328222

[13] Udge, Ganesh, Mahesh Mohite, Shubhankar Bendre, Yogeshwar Birnagal, and Disha Wankhede Mrs. "Statistical analysis for twitter spam detection." *I nternational Journal of Scientific Research in Science, Engineering and Technology* (2019): 624-29. https://doi.org/10.32628/IJSRSET1962170

[14] Santoshi, K. Ushasree, S. Sree Bhavya, Y. Bhavya Sri, and B. Venkateswarlu. "Twitter spam detection using naïve bayes classifier." In *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, pp. 773-777. IEEE, 2021. https://doi.org/10.1109/ICICT50816.2021.9358579

[15] Biyani, Yogita V., and Rahat Afreen Khan. "Spam detection in social media using machine learning algorithm." *Int J Res Appl Sci Eng Technol (IJRASET)* (2020). https://doi.org/10.22214/ijraset.2021.32832

[16] Shen, Hua, Xinyue Liu, and Xianchao Zhang. "Boosting social spam detection via attention mechanisms on twitter." *Electronics* 11, no. 7 (2022): 1129. https://doi.org/10.3390/electronics11071129

[17] Oyelakin, Akinyemi Moruff, A. O. Ameen, I. K. Ajiboye, I. S. Olatinwo, K. Y. Obiwusi, and T. S. Ogundele. "Evaluating the performance of heterogeneous and homogeneous ensemble-based models for twitter spam classification." *Innovative Computing Review* 2, no. 2 (2022). https://doi.org/10.32350/icr.0202.01

[18] Ainapure, Bharati S., Mythili Boopathi, Chandra Sekhar Kolli, and C. Jackulin. "Deep ensemble model for spam classification in twitter via sentiment extraction: Bio-inspiration-based classification model." *International Journal of Image and Graphics* 23, no. 04 (2023): 2350034. https://doi.org/10.1142/S0219467823500341

[19] Kaddoura, Sanaa, Ganesh Chandrasekaran, Daniela Elena Popescu, and Jude Hemanth Duraisamy. "A systematic literature review on spam content detection and classification." *PeerJ Computer Science* 8 (2022): e830. https://doi.org/10.7717/peerj-cs.830

[20] Dar, Momna, Faiza Iqbal, Rabia Latif, Ayesha Altaf, and Nor Shahida Mohd Jamail. "Policy-based spam detection of Tweets dataset." *Electronics* 12, no. 12 (2023): 2662. https://doi.org/10.3390/electronics12122662