Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 4 (No. 2), pp. 69-77, 2024
doi: 10.54569/aair.1549781
Published online: Dec 30, 2024

A Comparative Study of Machine Learning Classifiers for Different Language Spam SMS Detection: Performance Evaluation and Analysis

Samrat Kumar Dev Sharma [1,*]

[1] Department of Statistics, Jagannath University, Bangladesh

Abstract

With the continuous rise in the number of mobile device users, SMS (Short Message Service) remains a prevalent communication tool accessible on both smartphones and basic phones. Consequently, SMS traffic has experienced a significant surge. This increase has also led to a rise in spam messages, as spammers seek financial or business gains through activities like marketing promotions, lottery scams, and credit card information theft. Consequently, spam classification has become a focal point of research. In this paper, we explore the effectiveness of 11 machine learning algorithms for SMS spam detection, including multinomial Naïve Bayes, K-Nearest Neighbors (KNN), and Random Forest, among others. Utilizing datasets from UCI and Bangla SMS collections, our experimental results reveal that the multinomial Naïve Bayes algorithm surpasses previous models in spam detection, achieving accuracies of 98.65% and 89.10% in the respective datasets.

Keywords: *Spam SMS Detection, NLP, Machine Learning, Deep Learning, Naïve Bayes.*

1. Introduction

In today's digital age, mobile phones have become an indispensable part of daily life, with over 5.4 billion people worldwide having at least one mobile subscription, as reported by GSMA. This proliferation of mobile devices has led to a staggering number of mobile subscriptions surpassing the global population for the past several years, reaching over 8.58 billion subscriptions, according to the International Telecommunication Union (ITU). However, this unprecedented connectivity also presents challenges, one of which is the pervasive issue of spam SMS (Short Message Service) messages. Spam SMS messages, often utilized by fraudsters, have proliferated alongside the increasing usage of mobile devices. These unsolicited messages, ranging from marketing promotions to lottery scams and credit card information theft, pose significant risks to recipients, resulting in financial losses and privacy breaches. The prevalence of spam SMS is evident in statistics, with 39.3% of recipients being female and 59.4% male. Furthermore, Americans received a staggering 78 [1]billion automated spam texts in just the first half of one recent year alone. Given the detrimental impact of spam SMS on individuals and the increasing frequency of such messages, there is a pressing need for effective detection and mitigation strategies. Machine learning offers promising solutions in this regard, leveraging algorithms to analyze message content and classify messages as spam or legitimate (ham). In this study, we conduct a comparative analysis of 11 machine learning classifiers for spam SMS detection. By evaluating the performance of these classifiers using datasets from various sources, including the UCI repository and Bangla SMS collections, we aim to provide insights into the effectiveness of different algorithms in combating spam SMS. Through our research, we seek to contribute to the development of robust and reliable spam detection techniques, ultimately empowering users to better protect themselves from the scourge of spam SMS messages.

2. Related Work

In recent decades, researchers have explored various approaches and techniques to address this challenge, with a focus on leveraging machine learning algorithms for efficient detection. Gupta et al. [2] of a comparative study of spam SMS detection using 8 different classifiers including Support Vector Machine (SVM), Naive Bayes (NB), Decision Tree (DT), Logistic Regression (LG), Random Forest (RF), AdaBoost, ANN, and CNN. The test conducted [3] dataset that the authors show that the CNN and ANN have better accuracy compared to the other machine learning classifiers. The authors show that CNN is 98.25% and ANN is 98.00%, respectively. Xiaoxu Liu et al. [4] proposed 4 machine learning classifiers and deep learning transformers including LG, NB, RF, SVM, LSTM, CNN-LSTM, and spam Transformer of Spam SMS collection data set that the authors show logistic regression, Naïve Bayes, and SVM are better-performed machine learning classifiers. Here authors show accuracy is 98.56%, 98.38%, and 98.62%, respectively. Gadde et al. [5] using TF-IDF [6] word embedding technique and 6 classifiers algorithm. The authors proposed that the best algorithm is LSTM with an accuracy is 98.5%. Here authors also proposed RF and SVM whose accuracy achieved is 97.5% and 97% respectively on the SMS Spam Collection v.1 dataset. Then Suleiman et al. [7] using H2O framework to

*Corresponding author
E-mail address: samratkumardev2001@gmail.com

Received: 13/Sep/2024; Accepted: 28/Dec/2024.

achieve the highest accuracy of random forest algorithm. The authors claim accuracy is 97.7% respectively on the SMS Spam Collection v.1 dataset and using TF-IDF vectorizer algorithm to detect spam SMS and the authors get the accuracy for ham message is 99.46% and for spam accuracy is 95.90%. Haq et al. [8] proposed that logistic regression (LR) achieves a high accuracy of 99% respectively on the SMS Spam Collection v.1 dataset in detecting spam in mobile message communication, outperforming k-NN and decision tree (DT). Abayomi-Alli et al. [1] proposed that the highest accuracy is the Bi-LSTM model using SMS Spam Collection v.1 dataset that attained accuracy is 98.6% respectively. The authors also used SGD and Bayes Net models with accuracy, precision, recall, and F-measure of 96.8%, 96.9%, 91.7%, and 94.23%, respectively. Lim et al. [9] proposed the Cost-sensitive classifiers and Bayesian network to achieve the highest accuracy is 99.8\% respectively using the SMS Spam Collection v.1 dataset. Most researchers like [10, 11] and Himani Jain et al. [12] proposed the most efficient algorithms are Naïve Bayes and SVM. They are achieving accuracy of almost 97.93%, and 98.57% respectively. Tasmia et al. [13] proposed an ensemble approach to classify spam SMS Bengali text. Then Khan et al. [14] to proposed BERT Bangla SMS data set. and achieve accuracy 94% respectively. Lunna and Robert et al. [15, 16] proposed Bernoulli Naive Bayes model effectively detects and filters out spam in SMS texts with 96.63% accuracy, reducing security threats and fraud risks. Oyeyemi et al. [17] proposed the Naïve Bayes Alqahtani et al. [18] classifier + BERT model effectively detects and classifies SMS spam with a 97.31% accuracy and low false-positive rate, safeguarding users' privacy and assisting network providers. Yerima et al [6] proposed one class SVM classifier and Maqsood et al. [19] effectively detects and eliminates SMS spam with 98% overall accuracy and a low 3% false positive rate. Gupta et al. [20] using a voting classifier, a combination of four different classifiers including Gaussian NB, Bernoulli NB, Multinomial NB, and Decision Tree, provides more accurate spam detection than individual classifiers, with a mobile application to serve the purpose. The authors achieved 98.295% respectively of the SMS Spam Collection v.1 dataset.

3. Methodology

We started the workflow shown in **Figure 1**.at first data loading various sources, preprocessing and splitting data and applying the different classifier algorithms to train and test, and finally, evaluation and comparison to find the best models. Briefly discuss below

3.1. Dataset Description

In these experiments, we applied two different language datasets. The first dataset is the English SMS Spam Collection v.1 [3] dataset downloaded from the UCI repository. The dataset has 5572 rows and two columns. "v1" is the label (ham or spam) and "v2" is the messages.

The second dataset is Bangla SMS which was collected by personal survey. The researcher collected 504 data. This data set has two columns-target which label (ham or spam) and messages. The dataset is available as requested. It is shown **Table 1.**

Table 1. *Data Distribution between SMS Spam Collection v.1 and Bangla SMS*

	SMS Spam Collection v.1	Bangla SMS
Spam	747	217
Ham	4825	287
Total	5572	504

3.2. Data Preprocessing

The text-based data preprocessing pipeline involves several key steps. It begins with text cleaning to eliminate noise such as HTML tags, special characters, emoji, and punctuation. Tokenization then breaks down the text into smaller units, such as words or sentences. Next, stop words that do not significantly contribute to the text's semantics are removed. Finally, lemmatization is applied to reduce words to their base or root form, further streamlining the text data.

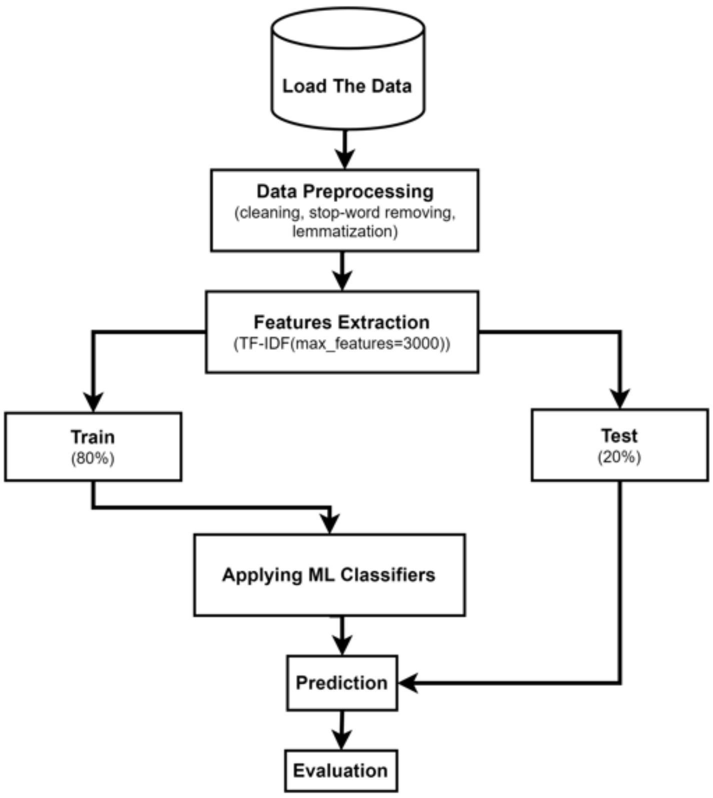

Figure 1. *Stepwise workflow*

After text data is preprocessed, it needs to be converted into a numerical format that machine learning algorithms can process. One common approach is to use the Term Frequency-Inverse Document Frequency (TF-IDF) technique, which assigns weights to words based on their frequency in the document and across the entire corpus. Each document is then represented as a vector, with each element corresponding to the TF-IDF score of a specific word. Transforming the text data into a TF-IDF matrix with a maximum of 3000 features allows us to capture the most relevant information while simultaneously reducing the dimensionality of the data. For the traditional machine learning approaches, the data is split into a training set (80%) and a test set (20%). This step is crucial for mitigating overfitting and handling imbalanced data. To ensure a representative distribution of classes in both the training and test sets, we employ a stratified splitting strategy. Stratified splitting preserves the relative proportions of different classes in both the training and test sets, thereby maintaining the balance between classes. By stratifying the data based on the target variable, we can effectively train the machine learning models on diverse samples while ensuring reliable evaluation of unseen data. This approach is particularly valuable when dealing with imbalanced datasets, where one class may be significantly more prevalent than others.

3.3. Machine Learning Classifiers

After completing text preprocessing, vectorization, and splitting, the data is ready for model classification. We apply 11 different classification models to the preprocessed and vectorized text data. These models include:

3.3.1. Naïve Bayes

Naïve Bayes is a simple probabilistic classifier based on Bayes' theorem with strong independence assumptions between features. The Multinomial Naïve Bayes (NB) classifier is a variant of the Naïve Bayes algorithm that is specifically designed for text classification tasks where the features are discrete and represent word counts or term frequencies.

$$\mathrm{P}(\mathrm{C_k}|\mathrm{doc}) = \frac{\mathrm{P}(\mathrm{C_k}) \times \prod_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{P}(\mathrm{w_i}|\mathrm{C_k})}{\mathrm{P}(\mathrm{doc})} \qquad (1)$$

Where:

- $P(C_k|doc)$ is the probability of classes C_k given the document.
- $P(C_k)$ is the prior probability of class C_k
- $P(w_i|C_k)$ is the probability of words w_i given class C_k
- $P(doc)$ is the probability of the document.

3.3.2. Logistic Regression

Logistic regression is a statistical method used for binary classification tasks. The logistic regression model applies a logistic function (also known as the sigmoid function) to linearly combine the features of the input data.

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_n X_n)}} \tag{2}$$

3.3.3. Support Vector Machine (*SVM*)

Support Vector Machine (SVM) is a powerful supervised learning algorithm used for classification tasks. Its main concept involves finding the hyperplane that best separates data points into different classes while maximizing the margin between the hyperplane and the closest data points (support vectors). This approach ensures robustness to noise and outliers in the data.

3.3.4. K-Nearest Neighbors (*KNN*)

The k-Nearest Neighbors (KNN) classifier is a simple, yet effective algorithm used for classification tasks. Its main concept involves assigning a class label to a data point based on the majority class among its K-nearest neighbors in the feature space. The choice of k determines the number of neighbors considered when making predictions.

3.3.5. Random Forest

The Random Forest classifier is an ensemble learning method that combines multiple decision trees to make predictions. It works by constructing a multitude of decision trees during training and outputting the mode of the classes (classification) or the mean prediction (regression) of the individual trees. Each tree in the forest is trained on a random subset of the training data, and during prediction, the results of all trees are aggregated to produce the final prediction. This approach helps reduce overfitting and improves the accuracy and robustness of the classifier.

3.3.6. Decision Tree

Decision tree classifiers are a type of supervised learning algorithm used for both classification and regression tasks. The main concept behind the decision trees is to recursively split the feature space into regions that are as pure as possible concerning the target variable. Each internal node of the tree represents a decision based on a feature, and each leaf node represents the predicted outcome. The decision tree is built by selecting the best feature to split the data at each node based on criteria such as Gini impurity or information gain. This process continues until a stopping criterion is met, such as reaching a maximum depth or a minimum number of samples per leaf. Decision trees are easy to interpret and visualize, making them popular for both exploration analysis and predictive modeling.

3.3.7. AdaBoost (Adaptive Boosting)

AdaBoost is an ensemble learning technique that combines multiple weak learners (often decision trees) to create a strong learner. It sequentially trains a series of weak learners, where each subsequent learner focuses more on the misclassified data points by giving them higher weights. The final prediction is made by aggregating the predictions of all weak learners, typically using a weighted majority voting scheme.

3.3.8. Bagging Classifier

Bagging (Bootstrap Aggregating) is an ensemble learning method that builds multiple base models (e.g., decision trees) on random subsets of the training data with replacement. Each base model is trained independently, and their predictions are combined using averaging (for regression) or voting (for classification) to make the final prediction.

3.3.9. ExtraTreesClassifier

Extra Trees (Extremely Randomized Trees) is an ensemble learning technique like Random Forest, where multiple decision trees are trained on random subsets of the training data. However, Extra Tree goes one step further by selecting random thresholds for each feature at each split point, resulting in even greater randomness and potentially reducing overfitting.

3.3.10. GradientBoostingClassifier

Gradient Boosting is an ensemble learning technique that builds multiple decision trees sequentially, with each tree aiming to correct the errors of its predecessor. In Gradient Boosting, each new tree is trained on the residual errors of the previous trees, gradually reducing the overall prediction error. The final prediction is obtained by summing the predictions of all trees.

3.3.11. XGBoost (Extreme Gradient Boosting)

XGBoost is an optimized implementation of Gradient Boosting that uses a more regularized model formalization to control overfitting and improve performance. It incorporates advanced features such as parallelized tree construction, hardware optimization, and efficient memory usage, making it one of the most popular and powerful gradient-boosting frameworks.

3.4. Model Training

All models were trained using the NVIDIA GeForce MX110 GPU, leveraging the Scikit-learn 1.4.2 library for machine learning tasks. This setup ensured efficient processing and utilization of computational resources during model training and evaluation.

4. Result and Analysis

4.1. Evaluation metrics

Evaluation metrics are used to assess the performance of the machine learning algorithm. It provides quantitative measures that help in comparing different models and selecting the most suitable one for a particular task. Common evaluation metrics are included as

Accuracy: It measures the proportion of correctly classified instances out of the total number of instances. It can be

$$\mathrm{Accuracy} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{3}$$

Where, TP (True Positive), TN (True Negative), FP (False Positive), FN (False Negative)

Precision: It measures the proportion of correctly predicted positive instances out of all instances predicted as positive. It can be formulated as:

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

Recall: It measures the proportion of correctly predicted positive instances out of all actual positive instances. It can be formulated as:

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

F1-Score: It is the harmonic means of precision and recall, providing a balance between the two metrics. It can be formulated as:

$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{6}$$

4.1. Experiment Results

We applied 11 different machine learning classifiers to two distinct SMS spam datasets. Our analysis revealed that the Multinomial Naïve Bayes classifier emerged as the top-performing model across both datasets, considering evaluation metrics such as accuracy, precision, recall, and f1-score, along with computational efficiency. We further improved its performance through hyperparameter tuning, as presented in **Table 2**. The experiment results for the SMS Spam Collection data are summarized in Table 3, while Table 4 presents the results for the Bangla SMS dataset.

Table 2. *Hyperparameters*

Model	Hyperparameters
SVC	Kernel: sigmoid, Gamma: 1.0
KNN	Default hyperparameters
Multinominal NB	Alpha: 0.01
DT	Max depth: 5
LG	Solver: liblinear, Penalty: l1
RF	Estimators: 50, Random state: 2
AdaBoost	Estimators: 50, Random state: 2
BG	Estimators: 50, Random state: 2
ET	Estimators: 50, Random state: 2
BG	Estimators: 50, Random state: 2
XGB	Estimators: 50, Random state: 2

Table 3. *Results of SMS Spam Collection v.1 dataset*

Algorithm	Accuracy %	Precision	Recall	F1-Score
MNB	98.65	0.9680	0.9236	0.9453
SVC	98.06	0.9826	0.8625	0.9186
RF	97.67	0.9652	0.8473	0.9024
BG	97.38	0.9482	0.8396	0.8906
AdaBoost	97.19	0.9047	0.8702	0.8871
ET	96.90	0.9541	0.7938	0.8666
XgBoost	96.61	0.8870	0.8396	0.8627
GB	95.45	0.9468	0.6793	0.7911
LG	95.16	0.9764	0.6335	0.7685
DT	92.94	0.8222	0.5648	0.6696
KNN	90.42	1.0000	0.2442	0.3926

Table 4. *Results of Bangla SMS dataset*

Algorithm	Accuracy%	Precision	Recall	F1-Score
MNB	89.10	0.8809	0.8604	0.8705
SVC	85.14	0.8500	0.7906	0.8192
RF	85.14	0.8333	0.8139	0.8235
BG	83.16	0.8250	0.7674	0.7951
AdaBoost	82.17	0.8048	0.7674	0.7857
ET	82.17	0.8205	0.7441	0.7804
XgBoost	82.17	0.8048	0.7674	0.7857
GB	79.21	0.7894	0.6976	0.7407
LG	78.21	0.7837	0.6744	0.7250
DT	70.29	0.7241	0.4883	0.5833
KNN	65.34	0.8333	0.2325	0.3636

4.3. Result Analysis

After analyzing the results, we propose selecting classifiers based on multiple factors including performance metrics and execution time. Models with higher accuracy, precision, recall, and F1-score are preferable, indicating better performance. Additionally, lower execution times are favored as they signify faster model training. From the summarized results in **Table 3**. Multinomial Naive Bayes emerges as the top performer, boasting high accuracy (98.65%), precision (96.80%), recall (92.37%), and F1-score (94.53%), alongside minimal execution time. Although SVC also demonstrates strong performance, it comes with a slightly longer execution time. While models like Extra Trees and Random Forest show competitive accuracies exceeding 97%, their longer execution times may hinder scalability. Overall, Multinomial Naive Bayes stands out for its exceptional performance and efficiency, though SVC remains a viable alternative depending on specific requirements. Considering the results from **Table 4**, the Multinomial Naive Bayes classifier exhibited the highest accuracy (89.11%), precision (88.10%), and F1-score (87.06%). Additionally, it achieved a respectable recall score of 86.05%. Moreover, it demonstrated the shortest execution time among all classifiers, taking only 0.015 seconds.

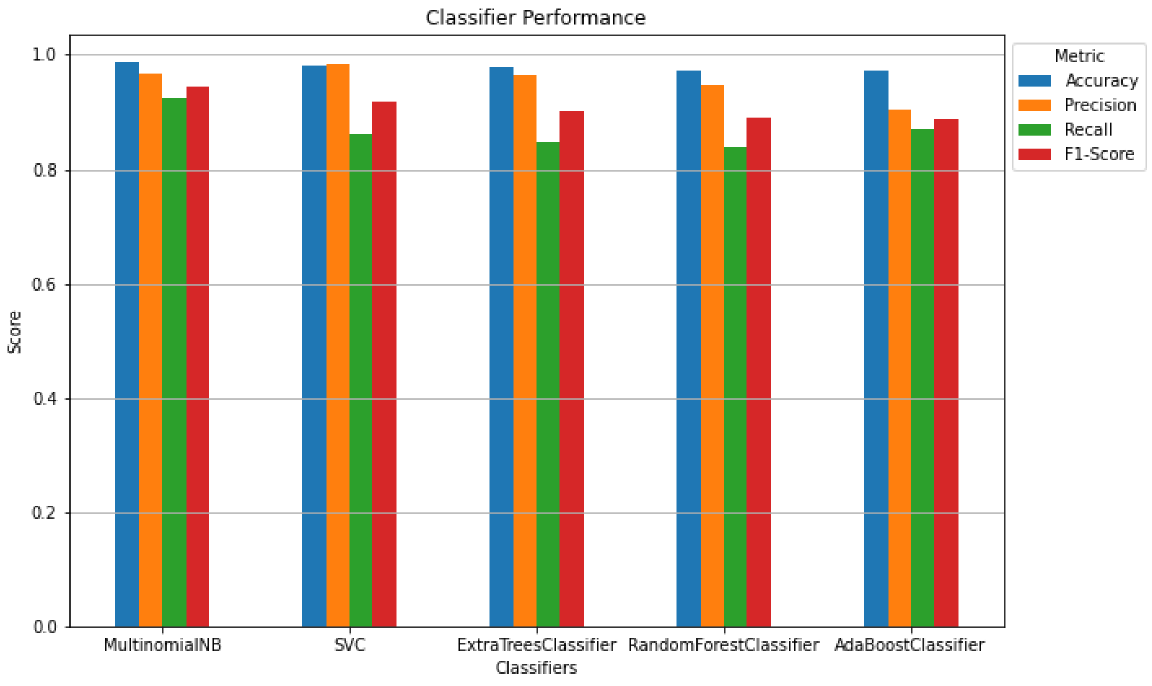

Figure 2. *Top five model performance SMS Collection v.1 dataset*

While the Support Vector Classifier (SVC) and Random Forest Classifier also delivered competitive performance, Multinomial Naive Bayes outperformed them in terms of both accuracy and execution time. The Bagging Classifier, AdaBoost Classifier, and Extra Trees Classifier followed suit, each showing comparable but slightly lower performance metrics compared to Multinomial Naive Bayes. It shows **Figure 2.** and **Figure 3**. for both datasets.

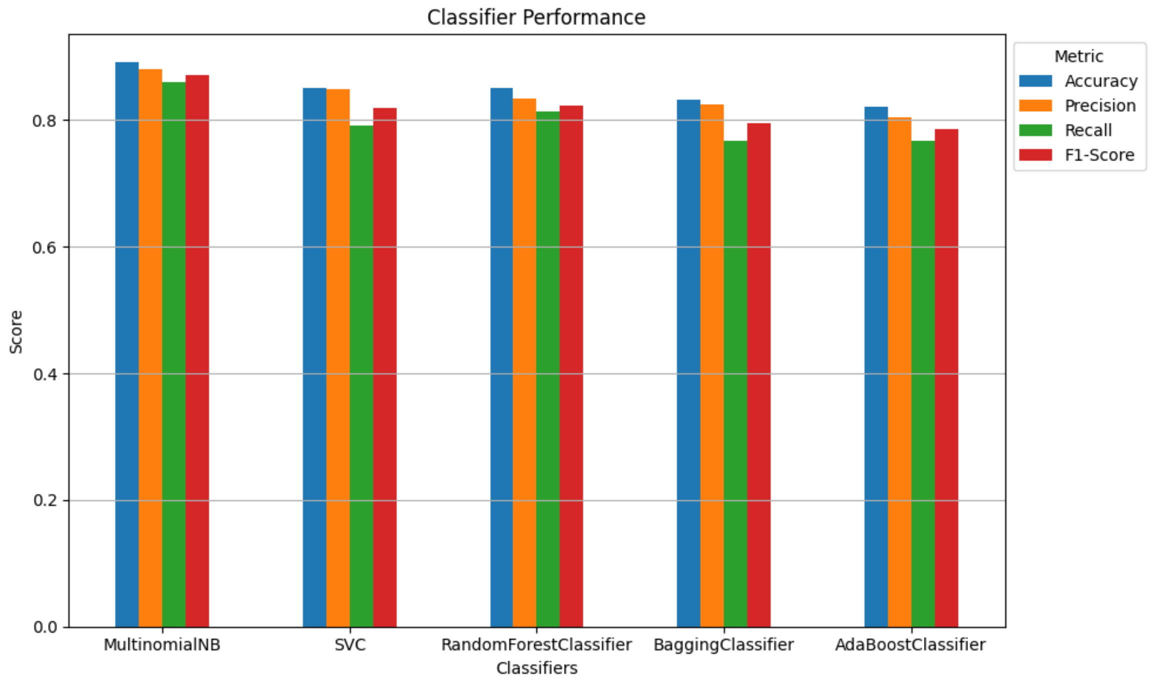

Figure 3. *Top five model performance Bangla SMS dataset*

On the other hand, the Logistic Regression, Gradient Boosting Classifier, Decision Tree Classifier, and K-Nearest Neighbors Classifier exhibited lower accuracy scores and longer execution times, making them less preferable choices in this scenario. Overall, Multinomial Naive Bayes stands out as the most efficient and effective classifier for two different datasets, offering a favorable balance between performance and computational cost. Thus, it is the recommended model for classification tasks on these datasets.

5. Conclusion and Future work

In conclusion, our analysis indicates that Multinomial Naive Bayes is the top-performing classifier for the given SMS spam datasets, exhibiting high accuracy and efficiency. However, further investigation could explore ensemble methods or deep learning techniques to potentially enhance classification performance. Additionally, incorporating more advanced feature engineering methods or exploring different text representation techniques may lead to improved model accuracy. Future work could also focus on deploying the selected classifier in real-world scenarios and evaluating its performance in a production environment.

References

[1] A. Alli and S. Misra, "A deep learning method for automatic SMS spam classification: Performance of learning algorithms on indigenous dataset," *Concurrency and Computation: Practice and Experience*, vol. 34, p. 34, 2022.

[2] S. D. Gupta, S. Saha and S. K. Das, "SMS spam detection using machine learning," in *Journal of Physics: Conference Series*, 2021.

[3] T. Almeida and J. Hidalgo, "SMS Spam Collection," 2011.

[4] X. Liu, H. Lu and A. Nayak, "A Spam Transformer Model for SMS Spam Detection," *IEEE Access*, vol. 9, pp. 80253-80263, 2021.

[5] 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)*, vol. 1, pp. 358-362, 2021.

[6] P. J. Yerima and S, "A comparative study of word embedding techniques for SMS spam detection," *2022 14th International Conference on Computational Intelligence and Communication Networks (CICN)*, pp. 149-155, 2022.

[7] D. Suleiman and G. Al-Naymat, "SMS spam detection using H2O framework," *Procedia computer science 113*, pp. 154-161, 2017.

[8] G. L. Haq, S. Nazir and H. U. Khan, "Spam Detection Approach for Secure Mobile Message Communication Using Machine Learning Algorithms," *Secur. Commun. Networks*, vol. 2020, pp. 8873639:1-8873639:6, 2020.

[9] L. P. Lim and M. M. Singh, "Resolving the imbalance issue in short messaging service spam dataset using cost-sensitive techniques," *Journal of Information Security and Applications*, vol. 54, p. 102558, 2020.

[10] E. Wijaya, G. Noveliora, K. D. Utami, Rojali and G. Z. Nabiilah, "Spam Detection in Short Message Service (SMS) Using Naïve Bayes, SVM, LSTM, and CNN," *2023 10th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*, pp. 431-436, 2023.

[11] E. Sankar, "Sms Spam Detection Using Machine Learning," *Interantional Journal Of Scientific Research In Engineering And Management*, 2023.

[12] Mahadev and H. Jain, "An Analysis of SMS Spam Detection using Machine Learning Model," *2022 Fifth International Conference on Computational Intelligence and Communication Technologies (CCICT)*, pp. 151-156, 2022.

[13] A. A. M. Tasmia, A. A. N. ,. Jidney and Z. M. A. M. Haque, "Ensemble Approach to Classify Spam SMS from Bengali Text," in *Springer Nature*, kolkata, 2023.

[14] F. Khan, R. Mustafa, F. Tasnim, T. Mahmud, M. S. Hossain and K. Andersson, "Exploring BERT and ELMo for Bangla Spam SMS Dataset Creation and Detection," in *2023 26th International Conference on Computer and Information Technology (ICCIT)*, 2023.

[15] R. G. d. Luna, V. C. Magnaye, R. A. L. Reaño, K. L. Enriquez, D. Astorga, T. Celestial, A. M. Española, B. A. Lanting, D. Mugar, M. Ramos and J. Redondo, "A Machine Learning Approach for

Efficient Spam Detection in Short Messaging System (SMS)," *TENCON 2023 - 2023 IEEE Region 10 Conference (TENCON),* pp. 53-58, 2023.

[16] R. G. d. L. Redondo, V. C. Magnaye, R. A. L. Reaño, K. L. E. a. D. Astorga, T. Celestial, A. M. Española, B. A. Lanting, D. Mugar, M. Ramos and Jenjazel, "A Machine Learning Approach for Efficient Spam Detection in Short Messaging System (SMS)," *TENCON 2023 - 2023 IEEE Region 10 Conference (TENCON),* pp. 53-58, 2023.

[17] Ojo and D. A. Oyeyemi, "SMS Spam Detection and Classification to Combat Abuse in Telephone Networks Using Natural Language Processing," *Journal of Advances in Mathematics and Computer Science,* 2023.

[18] S. Alghazzawi and D. Alqahtani, "A survey of Emerging Techniques in Detecting SMS Spam," *Transactions on Machine Learning and Artificial Intelligence,* 2019.

[19] U. M. Kundi, S. Rehman, T. Ali, K. Mahmood and T. Alsaedi, "An Intelligent Framework Based on Deep Learning for SMS and e-mail Spam Detection," *Appl. Comput. Intell. Soft Comput.,* vol. 2023, pp. 6648970:1-6648970:16, 2023.

[20] M. Gupta, A. Bakliwal, S. Agarwal and P. Mehndiratta, "A comparative study of spam SMS detection using machine learning classifiers," in *IEEE,* 2018.

[21] T. A. H. Almeida and Y. A. Jos'e Maria G, "Contributions to the study of SMS spam filtering: new collection and results," in *Association for Computing Machinery,* New York, NY, USA, 2011.

[22] Bashar and S. Yerima, "Semi-supervised novelty detection with one class SVM for SMS spam detection," *2022 29th International Conference on Systems, Signals and Image Processing (IWSSIP),* Vols. CFP2255E-ART, pp. 1-4, 2022.

[23] 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),* vol. 1, pp. 358-362, 2021.

[24] E. W. Nabiilah, G. Noveliora, K. D. Utami, Rojali and G. Zain, "Spam Detection in Short Message Service (SMS) Using Naïve Bayes, SVM, LSTM, and CNN," *2023 10th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE),* pp. 431-436, 2023.

[25] P. K. Roy, J. P. Singh and S. Banerjee, "Deep learning to filter SMS Spam," *Future Generation Computer Systems,* vol. 102, pp. 524-533, 2020.

[26] S. Yadav and A., "Mobile SMS Spam Filtering for Nepali Text Using Naïve Bayesian and Support Vector Machine," *International Journal of Intelligent Systems,* vol. 04, pp. 24-28, 2014.