Comparison of Multiple Machine Learning Approaches and Sentiment Analysis in Detection of Spam on Different Type of Data

A. N. M. Sajedul Alam[1], Shifat Zaman[1], Zawad Alam[1], Junaid Bin Kibria[1] Arnob Kumar Dey[1], Mohammed Julfikar Ali Mahbub[1], Motahar Mahtab[1], Annajiat Alim Rasel[1]

[1] Department of Computer Science and Engineering, School of Data and Sciences (SDS), BRAC University, 66 Mohakhali, Dhaka 1212, Bangladesh
{a.n.m.sajedul.alam, shifat.zaman, zawad.alam, junaid.bin.kibria, arnob.kumar.dey, mohammed.julfikar.ali.mahbub, md.motahar.mahtab}@g.bracu.ac.bd, annajiat@bracu.ac.bd

Abstract. Currently, all of our communications are made through various electronic communication mediums. While it has made communication between people from different parts of the world very easy, things like spamming have made life difficult for many people. Spammers are found in almost every electronic communication platform like email, mobile SMS, social networking sites, etc. So, with time detecting spam messages and filtering them out from our important messages have become more and more important. For many years, Natural Language Processing (NLP) researchers have proposed different techniques to detect spam messages. In this paper, our objective is to detect spam messages in a dataset using vectorization along with various machine learning algorithms and compare their results to find out the best classifier for detecting spam messages.

Keywords: Machine Learning, Natural Language Processing, Spam Detection

1 Introduction

In the modern world, where electronic communication is a part of our day-to-day activities, spamming has become a major problem. It affects users of various major platforms like email, mobile SMS, social networking sites, etc. Nowadays, most email clients have their own spam filters, which also block out specific spam email addresses. However, spam messages still make it to the end-users either through bypassing these filters or mobile messages or social network messaging applications. When the question is about filtering out spam messages based on their internal texts, NLP (Natural Language Processing) is the biggest tool we can use. NLP has been

used for a long time to process the information inside a message along with various machine learning algorithms to filter out spam messages. In this paper, we have used NLP with seven different machine learning algorithms to classify spam and non-spam (ham) messages. We have also compared the results and accuracy of each technique in order to identify which can be the best solution to categorize spam messages.

Spam messages have been there in our life for a long time. Thousands of users have suffered personal and financial losses due to these unwanted messages. To solve these problems, researchers have introduced different techniques from time to time. Machine Learning and more specifically NLP researchers brought out advanced solutions to these problems. N. Govil, K. Agarwal, A. Bansal, and A. Varshney proposed a method in their paper "A Machine Learning based Spam Detection Mechanism"[1] where they created a dictionary of spam words and then used Naive Bayes Classification to detect spam messages. N. Kumar, S. Sonowal, and Nishant [2] proposed a method where they first tokenized the messages, extracted features from them, and then trained them using various machine learning techniques. However, as the number and types of spam messages are growing, it is important to find out the best solution for detecting and classifying spam messages.

In this paper, our objective is to detect spam messages using vectorization and seven machine learning algorithms and compare the results of these classifiers. We will be using TFIDF vectorizer for vectorization as well as seven classifiers - Random Forest, XGBoost, Naive Bayes, Logistic Regression, SVC, Cat boost, and K nearest neighbor. This paper first discusses several techniques and researches for spam detection, then introduces our technique for solving the problem and comparison of results.

2 Literature Review

Processing of Natural language and mining of opinion fields are used to cross the barriers in every single language due to the fact that analysis of opinion is dependent on language. In 2020, Najed et al. gave all their attention to a feature set to develop which can be utilized along with classifiers of numerous types as an input of a reliable source for detecting opinion spam in the Persian language [3]. They used some of the elements discovered during their research that they made while reviewing detection of spam in the English language, modified the definition along with some of the usability to fit the language in Persian, plus added additional novel features as a result. Their investigations show that the classifiers with the best results for detecting opinion spam in Persian are AdaBoost and Decision Tree, getting accuracy percentages of 98.7 and 98, respectively. In relation to the goal of detecting opinion spam, the robustness of their enlarged feature set was also tested by comparing it with other feature sets.

The internet's rapid expansion has made us reliant on it for the majority of our activities. E-commerce is one such rapidly expanding field. Hundreds of e-commerce sites exist today, with millions of products available. These websites allow users to leave feedback on the product or service they received. These reviews can be a valuable source of data for service providers that are analyzing sales or making decisions. It can also help potential customers decide whether or not to purchase a product. These reviews, however, are only beneficial if they are accurate. The rapid expansion of e-commerce has resulted in fierce competition among businesses and brands. As a result, some parties attempt to post phony reviews on a competitor's website in order to boost the popularity of a product or brand or to disparage it. In their research, Aishwarya et al. in 2019 used the Naive Bayes algorithm to address the issue to put a score on the reviews and give labels to them, thereby distinguishing the real and fake reviews [4].

Peng, Q., and Zhong, M. attempted to integrate sentiment analysis with spam review detection in [5]. The researchers came up with three projects to focus on. To calculate the sentiment score, first, develop a sentiment lexicon and then utilize a shallow dependency parser. The following step is to construct a set of discriminant rules. The final aim is to use a time series technique to develop a strategy for detecting spam reviews. SentiWordNet and MPQA were utilized for sentiment analysis. MPQA achieves a greater classification accuracy than SentiWordNet, which obtains a 61.4 percent accuracy. The results for Spam Review Detection are all above 0.65, indicating a high level of agreement. As a result, it is clear that evaluators' judgements are consistent and effective. Furthermore, the sentiment score approach has a higher accuracy than the rating method. They discovered that certain reviews' ratings conflict with the sentiment stated in natural language text. Because it ignores negation terms in the reviews, the word counting method has a low accuracy.

3 Proposed Methodology

It consists of 4 subsections: Section A deals with data collection, Section B describes the dataset we used, Section C deals with the Vectorization and Section D describes the modeling and Evaluation

A. Dataset Collection: The dataset that we have used here in order to train our model has been collected from the UCI[6] datasets, which is a public dataset that contains SMS labeled messages.

B. Dataset Description: Our dataset includes 5574 SMS phone messages in English. Each message is tagged as either SPAM or HAM based on its legitimacy. We have labeled the message texts as 'message' and their tags as 'label' for further usage. In our dataset, there are 4825 messages that are labeled as 'ham' or legit, on the other hand, there are 747 messages that are labeled as spam.

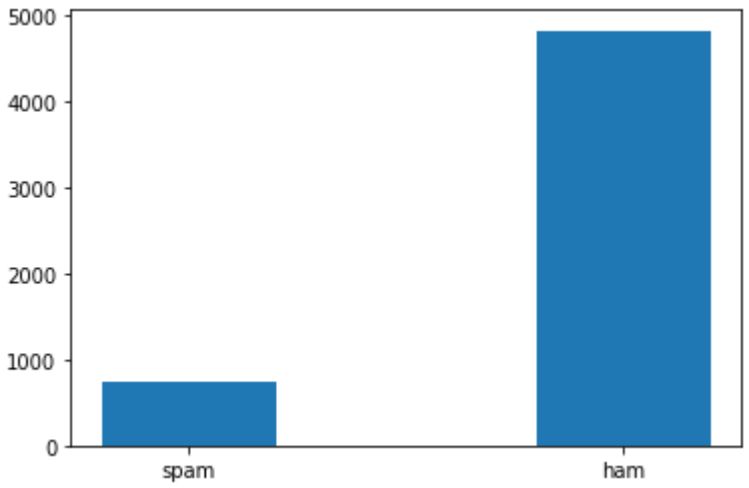

Fig. 1. A bar chart of spam and ham

C. Vectorization: It is the base building block of our NLP pipeline. We have used TfidfVectorizer in order to transform our message texts to feature vectors. These transformed feature vectors will be used in various estimators. TF-IDF is an abbreviation for the term frequency-inverse document frequency. In the field of data retrieval, the TF-IDF weight represents a commonly used term. The importance of a word in a text may be gauged using the weight assigned to it in a corpus. The significance of a word is related to how many times it appears in the manuscript. Aiming to reduce the influence of tokens that appear often but are experimentally less informative than those that only appear in a tiny percentage of a corpus, the TF-IDF method replaces raw frequency counts of tokens in a text[7].

The TF-IDF weight is made up of two components. The termed frequency (TF) is computed by dividing the total number of words in the document by the number of occurrences of that word. The Inverse Document Frequency

(IDF) is calculated by dividing the logarithm of the number of documents in the corpus by the number of documents containing that particular phrase.

TF (Term Frequency): It calculates how frequently a term occurs in a document. The term frequency is often divided by the document length,

> TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document).

IDF (Inverse Document Frequency):, It measures the importance of a term. All terms are given the same importance while calculating TF. But, some common terms like 'of', 'is', 'that' can occur many times in a document but have little importance. So, we need to weigh down these less important frequent terms and scale-up rare and important ones. IDF is calculated in the following way -

> IDF(t) = log_e(Total number of documents / Number of documents with term t in it).

D. Modeling and Evaluation

i) Classification Model

In this stage after vectorization, we have our features represented as vectors. Now we can train our classifier for spam detection. First, we shuffled the dataset and then split the data into train and test data. We took 70% training data and 30% testing data. We have used a total of 7 classification algorithms for our research and then observed the results. Classification algorithms that we used here are - Random Forest, XG-Boost, SVC, Naive Bayes, Logistic Regression, Cat Boost, and K-Nearest Neighbor. A brief description of seven algorithms is narrated below.

A random forest is one kind of machine learning method. Interestingly, this method can be used to resolve regression as well as classification issues. It is made up of several decision trees. It is a strategy that combines a large number of classifiers to solve complicated issues.

XGBoost is a gradient boosting machine implementation that is designed to tax the computing resources of boosted tree algorithms. It is designed for speed and model performance. By maximizing the distance between the sample points and the hyperplane, the Support Vectors Classifier attempts to discover the optimal hyperplane to partition the different classes. A Naive

Bayes classifier is a probabilistic machine learning model for solving classification issues. The classifier's challenge is centered on the Bayes theorem.

In its most basic form, logistic regression is a statistical model that employs a logistic function to represent a binary dependent variable; however, there are various more complex forms available.[8].

The K-NN algorithm is a supervised machine learning technique that assumes similarity between new and old cases/data and allocates the new example to the category that is most similar to the existing categories.

CatBoost is an open-source machine learning method that generates cutting-edge results without requiring extensive data training and provides powerful out-of-the-box support for more descriptive data types.

ii) Confusion Matrix
The confusion matrix indicates performance measurement in classification issues. It depicts a model's perplexed state during prediction. Depending on the problem statement, the outcome can be two or more than two.

iii) Evaluation Metrics
Evaluation metrics are used to assess the quality of a statistical or machine learning model. Precision, F1-Score, and Recall are the metrics we're looking at. We will use these evaluation metrics to seven algorithms to determine which one performs the best.

4 Result Analysis

As previously stated, we vectorized our dataset and trained our model using seven distinct machine learning algorithms. We tested our classifiers using 30% of our dataset. For each classifier outcome, we built a confusion matrix, roc curve, and precision vs recall curve to examine the findings. Finally, the scores of each classifier were displayed in a single table.

i) Random Forest: We generated a confusion matrix based on predictions made by the Random Forest classifier on our test dataset (Fig 2). Predictions are given on X-axis whereas actual results are plotted on Y-axis. From the confusion matrix, we can see, out of all spam messages, a total of 195 messages were correctly classified

whereas 29 messages were wrongfully classified as ‘ham’. All the ham messages were correctly classified.

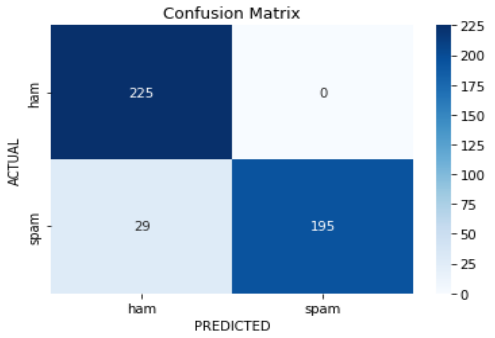

Fig. 2. Confusion matrix for Random Forest

ii) XGBoost: We generated a confusion matrix based on predictions made by the XGBoost library (which uses Gradient boosting) on our test dataset (Fig 3). For XGBoost, from the confusion matrix, we can see, out of all spam messages, a total of 196 messages were correctly classified whereas 28 messages were wrongfully classified as ‘ham’. 224 messages were correctly classified except for one message.

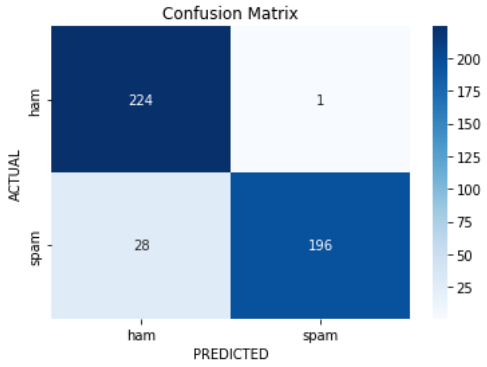

Fig. 3. Confusion matrix for XGBoost

iii) SVC: For SVC, from the confusion matrix, we can see, out of all spam messages, a total of 204 messages are correctly classified (fig 4). 20 messages are

wrongfully classified as 'ham'. For ham messages, 220 messages are correctly classified and 5 messages are wrongfully classified.

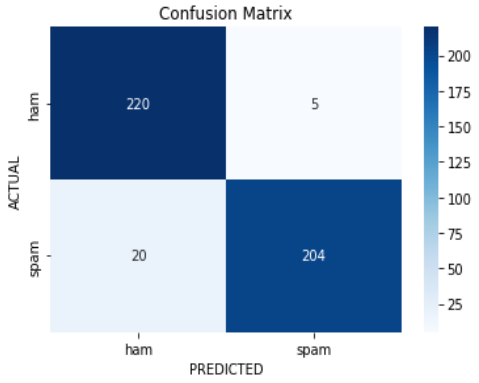

Fig. 4. Confusion matrix for SVC

iv) Naive Bayes: From the confusion matrix, we can see, out of all spam messages, a total of 210 messages are correctly classified whereas 14 messages are wrongfully classified as 'ham'. For ham messages, 217 messages are correctly classified and 8 messages are wrongfully classified (fig 5).

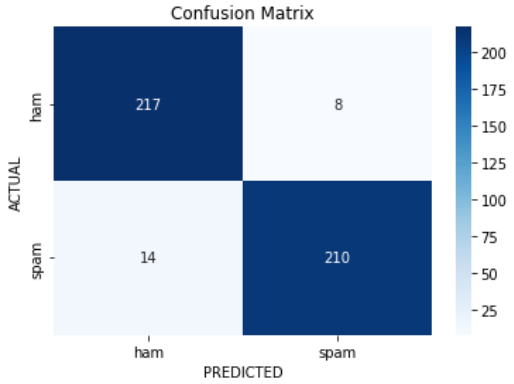

Fig. 5. Confusion matrix for Naive Bayes

v) Logistic Regression: From the confusion matrix, we can see, out of all spam messages, a total of 210 messages are correctly classified whereas 14 messages are wrongfully classified as 'ham'. For ham messages, 217 messages are correctly classified and 8 messages are wrongfully classified (fig 6).

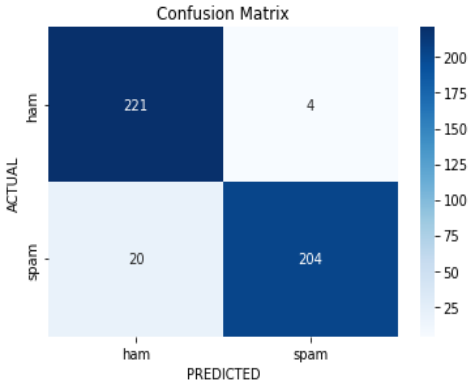

Fig. 6. Confusion matrix for Logistic Regression

vi) CatBoost: From the confusion matrix, we can see, out of all spam messages, a total of 195 messages are correctly classified whereas 29 messages are wrongfully classified as 'ham'. All the ham messages were correctly classified (fig 7).

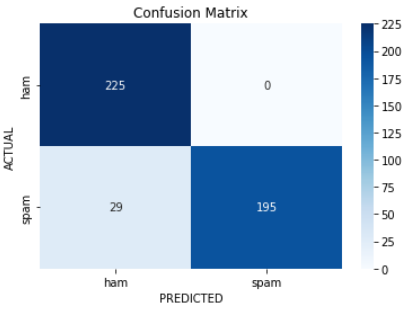

Fig. 7. Confusion matrix for CatBoost

vii) KNN: From the confusion matrix, we can see, out of all spam messages, a total of 203 messages are correctly classified whereas 21 messages are wrongfully classified as 'ham'. For ham messages, 219 messages are correctly classified and 6 messages are wrongfully classified (fig 8).

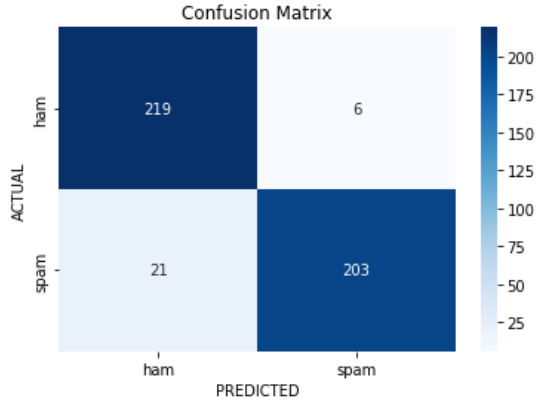

Fig. 8. Confusion matrix for KNN

Algorithms	Train F1_Score	Test F1_Score	Overfitting	Accuracy
Random Forest	1	0.935	True	0.935412
XGBoost	0.964	0.935	True	0.935412
Naive Bayes	0.979	0.951	True	0.951002
SVC	0.99	0.944	True	0.944321
Logistic Regression	0.98	0.946	True	0.946548
CatBoost	0.998	0.935	True	0.935412
KNN	0.945	0.94	True	0.939866

Table 1. .The overall score for evaluation metrics on six algorithms focusing Train F1, Test F1, Overfitting, and Accuracy scores

From table 1 above, we find out the train f1_score and test f1_score for the seven algorithms. For train f1_score, random forest shows the best value with 1 and for test f1_score, naive Bayes shows the best value with 0.951. We also find out the overfitting for all algorithms which is true. There is not much difference in accuracy scores among all the algorithms. Among them, Naive Bayes shows the best accuracy score with 0.951002.

Algorithms	ROC Area	Precision	Recall	F1
Random Forest	0.991389	1	0.870536	0.930788
XGBoost	0.982133	0.994924	0.875	0.931116
Naive Bayes	0.987986	0.963303	0.9375	0.950226
SVC	0.993938	0.976077	0.910714	0.942263
Logistic Regression	0.988958	0.980769	0.910714	0.944444
CatBoost	0.986796	1	0.870536	0.930788
KNN	0.979504	0.971292	0.90625	0.937644

Table 2. The overall score for evaluation metrics on six algorithms focusing ROC Area, Precision, Recall, and F1 scores.

From **Table 2** above, we find out ROC area, precision, f1 score, and recall for seven algorithms. For ROC and precision, random forest shows the highest score with 0.991389 and precision shows the highest score with 1 respectively. On the other hand, for recall and f1 score, Naive Bayes shows the best performance score with 0.9375 and 0.950226 respectively.

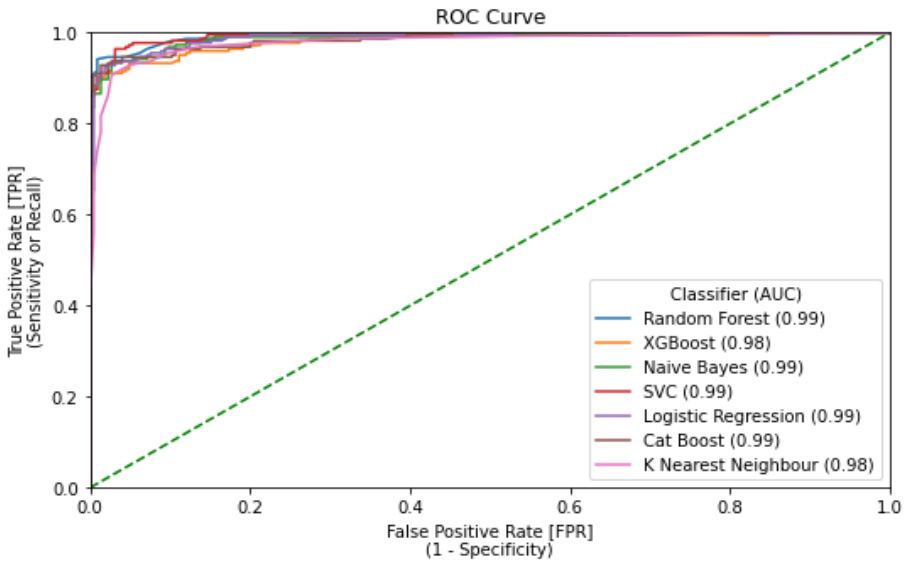

Fig. 9. ROC Curve

The ROC curve is a classification task performance statistic with variable threshold values. If the AUC is close to zero, it has the poorest metric of separability. In addition, if the AUC is 0.5, the model has no class separation capacity. Depending on the threshold, the AUC for all algorithms is 0.99 or 0.98. As a result, it performs well, and the model can distinguish between positive and negative classes.

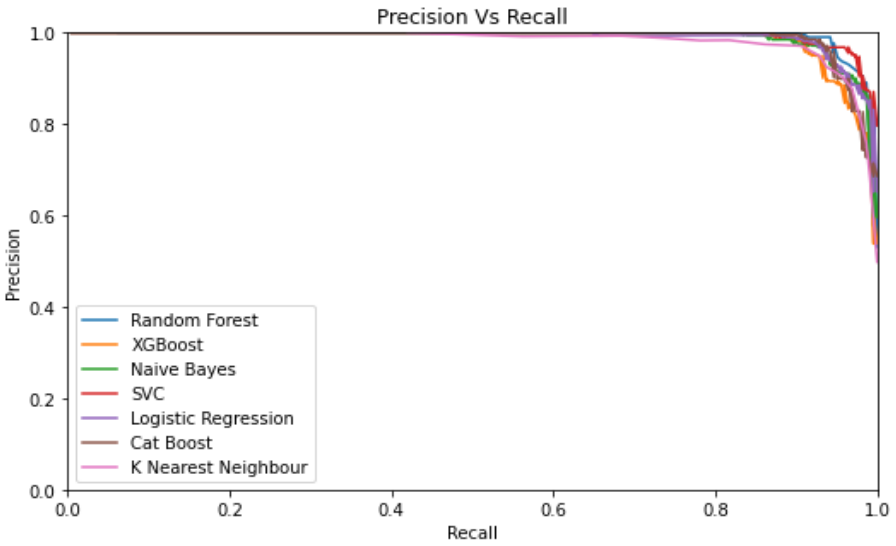

Fig. 10. Precision vs Recall Curve

The precision-recall curve is used to evaluate the efficacy of binary classification algorithms. It is typically used in situations where the courses are extremely uneven. The picture above depicts the precision vs recall curve for classifiers. A PR-curve with a PR-AUC of one span horizontally from the top left corner to the top right corner and straight down to the bottom right corner. It can be demonstrated that the curves outperform all of the classifiers.

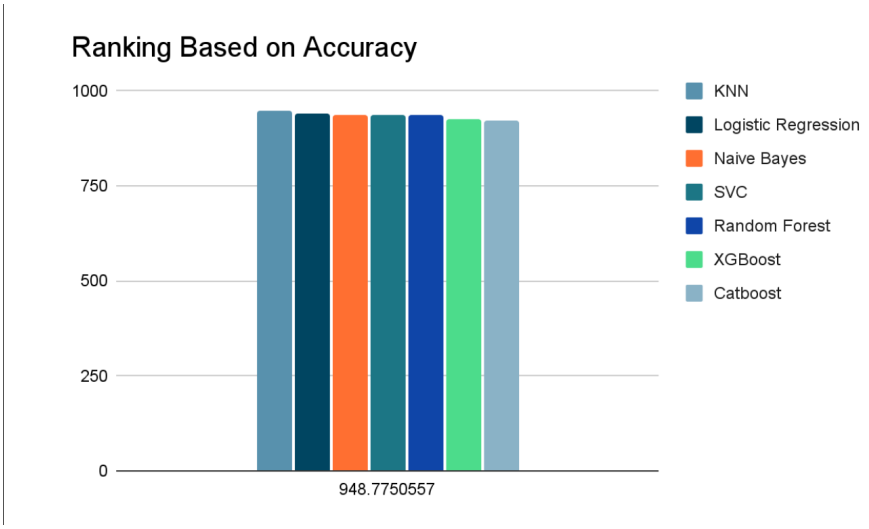

Fig. 11. A ranking of algorithms based on only accuracy scores

Based only on accuracy ratings, Catboost (0.922) performs the lowest, and K-nearest neighbor (0.948) performs the best; also, Logistic regression is the second-best performer (0.942). SVC and Naive Bayes, two other approaches that performed at least as well as Catboost, were both in the same accuracy range as Catboost (0.937). This bar chart (Fig 11) clearly shows that KNN outperforms all of the other approaches in the list. It can almost perfectly detect the majority of spam.

5 Conclusion

People communicate through messages, and millions of smartphone users send and receive countless messages every day. Unfortunately, because of the lack of suitable message filtering techniques, that kind of communication is unsafe. Spam is an example of such instability, as it compromises the security of message communication. Spam detection is a significant problem in the modern message exchange. As a result, message transmission is not secure. To combat this issue, an effective and exact mechanism for detecting spam in message transmission is required. In this study, we used TFIDF vectorizer for vectorization and also used seven classifiers, which are Random Forest, XGBoost, Naive Bayes, Logistic Regression, SVC, Cat boost, and K nearest neighbor. After using several classifiers, it is seen that K nearest neighbor(KNN) provides the best accuracy than other machine learning algorithms.

In the future, we want to implement the BERT technique instead of the TFIDF vectorizer for more precise data preprocessing. We will also try to use other classifier models to show better scores.

References

1. N. Govil, K. Agarwal, A. Bansal and A. Varshney, "A Machine Learning based Spam Detection Mechanism," 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC), 2020, pp. 954-957, doi: 10.1109/ICCMC48092.2020.ICCMC-000177.
2. N. Kumar, S. Sonowal and Nishant, "Email Spam Detection Using Machine Learning Algorithms," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), 2020, pp. 108-113, doi: 10.1109/ICIRCA48905.2020.9183098.
3. Aagte, A. A., Vrushali, W., Vishwakarma, P., & Kamble, S. (2019, May). Spam Detection using Sentiment Analysis. In *2020 10th International Conference on Computer and Knowledge Engineering (ICCKE)* (pp. 209-214). IEEE.
4. Nejad, S. J., Ahmadi-Abkenari, F., & Bayat, P. (2020, October). Opinion Spam Detection based on Supervised Sentiment Analysis Approach. In *2019 International Journal of Emerging Technologies and Innovative Research* (pp. 228-232). JETIR.
5. Peng, Q., & Zhong, M. (2014). Detecting Spam Review through Sentiment Analysis. *J. Softw.*, *9*(8), 2065-2072.
6. Tiago A.A (2012,June 22nd), UCI Machine Learning Repository: SMS Spam Collection Data Set. Retrieved from https://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection
7. Gaydhani, A., Doma, V., Kendre, S., & Bhagwat, L. (2018). Detecting hate speech and offensive language on twitter using machine learning: An n-gram and tfidf based approach. *arXiv preprint arXiv:1809.08651*.
8. Saif, M. A., Medvedev, A. N., Medvedev, M. A., & Atanasova, T. (2018, December). Classification of online toxic comments using the logistic regression and neural networks models. In *AIP conference proceedings* (Vol. 2048, No. 1, p. 060011). AIP Publishing LLC.