Improve the Sentiment of Bengali Language Texts with Stopword Removal

G. Mostafa
Department of Computer Science and Engineering
United International University
Dhaka, Bangladesh
gmostafa1210@gmail.com

Md Sabbir Ahamed
Lamar University
Beaumont, Texas, USA
mahamed1@lamar.edu

Md Kishor Morol
Cornell University
Ithaca, New York, USA
kishor@elitelab.ai

Abstract

Many sentiment analysis research studies have been conducted, mainly based on the English language. To clean up the English language dataset, several applications have been developed with built-in functions to facilitate the research work. One such function is the Stopword function. It is frequently used to eliminate unnecessary words from sentences to preprocess datasets in various languages. Similarly, research is being carried out to analyze sentiments using the Bengali language. Bengali language is also filled with unnecessary and noisy words that do not express feelings. The objective of this research is to demonstrate that the removal of stopwords can significantly enhance the accuracy of sentiment analysis in Bengali language texts. In this paper, a Stopword function was implemented to remove unwanted words from a dataset of Bengali sentences, and different machine learning algorithms were used for classification. The outputs were compared before and after the implementation of the Stopword function, resulting break a hand accuracy.

CCS Concepts

• **Computing methodologies**; • **Machine learning; Supervised learning; Natural language processing; Information extraction.**;

Keywords

Stopword Removal, Tokenization, Sentiment Analysis, Bengali Language, Natural Language Processing, NLP, Machine Learning, Supervised learning, Intelligent Systems

ACM Reference Format:

G. Mostafa, Md Sabbir Ahamed, and Md Kishor Morol. 2024. Improve the Sentiment of Bengali Language Texts with Stopword Removal. In *3rd International Conference on Computing Advancements (ICCA 2024), October 17, 18, 2024, Dhaka, Bangladesh.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3723178.3723233

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
ICCA 2024, Dhaka, Bangladesh
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-1382-8/2024/10
https://doi.org/10.1145/3723178.3723233

1 Introduction

In today's world, it is customary to communicate and access information through the Internet. Social media platforms have become widely used to expand global interactions and relationships. People can express their emotions about any content on Facebook, YouTube, Twitter, and other social networks, regardless of the language used. To handle natural language in various applications, we use pattern recognition and machine learning (ML) techniques [21]. An intelligent automated software capable of detecting the sentiment of a language would be beneficial, enhancing user experiences and reducing the need for manual intervention. Adopting a conversational spoken manner challenges language conventions, resulting in a casual tone. With the increase in data and frequent users, NLP is now focused on understanding social media content.

Sentiment analysis (SA) is a method that has already been developed and is being used in various sectors to analyze and determine the author's emotions in different languages [25]. It has become an important research topic in today's world, particularly in analyzing data in several languages. The aim is to apply this method to analyze the sentiments of Bengali writers through their sentences. With the Internet being widely used throughout the country and online platforms becoming increasingly popular among people in their daily lives, it is necessary to create datasets based on the analysis of emotions and feelings of Bengali people in various domains [17]. This paper presents a dataset that contains Bengali lines that express the perceptions and opinions of ordinary people.

When conducting SA, removing stopwords can help to emphasize the more meaningful words that express sentiment. This shift in focus toward informative and sentiment-bearing words enhances the accuracy and efficiency of SA [13]. In this study, we collected a dataset of text samples written in Bengali. The samples were then labelled with six different emotional categories. Additionally, we created a custom list of stopwords specifically for Bengali writing. Using this list, we developed a method to remove stopwords from the dataset. We then used various ML techniques to categorize the Bengali text samples by emotions. We also performed experiments to compare the accuracy before and after the removal of the stopword. The results of our study showed significant differences in the accuracy of several ML algorithms when stopword removal was implemented. This highlights the importance of preprocessing techniques in improving the classification performance for emotional text analysis in Bengali. As we work with the Bengali language, some examples of Bengali sentence inputs are described below. If the user gives an input like তাকে নিলে তার ভাল হবে, then the system will analyze this sentence and give a prediction. However, the

sentence can be predicted by the system only based on নিলে ভাল হবে. So, implementing the Stopword function, we remove the words তাকে and তার, making the sentence more efficient and simpler to train the system with fewer parameters. The experimental findings we obtained highlight contributing to a better understanding of effectiveness of stopword elimination in Bengali text processing and emphasize its importance in enhancing the accuracy or categorization for emotional text analysis tasks. The following are significant contributions of this SA research work:

- Manually create a dataset consisting of six different sentiments in the Bengali language and annotate them.
- Prepare a list of words to implement the Stopword function.
- Apply various ML algorithms and demonstrate the accuracies both before and after implementing the developed Stopword Function.

2 RELATED WORKS

The field of SA is vast, with a plethora of research work. The paper [7] conducted a study using neural network models to apply SA to Twitter data on COVID-19 vaccinations. The objective was to identify the predominant sentiments expressed by Twitter users. The paper discusses data collection, preprocessing, model training, and evaluation metrics. This research provides valuable information on Twitter on the public's sentiment toward COVID-19 vaccines. Similarly, the paper [20] conducted a study that analyzed sentiment in Bengali Facebook posts, utilizing advanced data collection and SA techniques. An overview of emotions, opinions, and attitudes among Bengali-speaking users on social media is discussed here, which can be helpful for marketers, legislators, and researchers. The research enhances our understanding of the digital environment in Bengali-speaking areas and offers a strong foundation for studying sentiment in Bengali Facebook posts.

To overcome the difficulty of identifying false news in Bangla, paper [8] offers techniques to enhance model performance. The two main tactics that the writers concentrate on are managing class imbalance and using model stacking. To ensure a more balanced dataset, methods like oversampling and undersampling are used to address class imbalance, a prevalent problem in false news identification. Combining many machine learning models is known as "model stacking," and it improves prediction accuracy. The work highlights the potential of these approaches for applicability in additional low-resource languages by demonstrating how much they boost the efficacy of false news detection systems in Bangla.

The paper [22] presents a new method for identifying Bengali spoken digits using deep learning (DL) techniques, specifically CNN. The authors experimented with a dataset of Bengali spoken digits and measured the accuracy of their method's classification. The research provides valuable insights for voice recognition systems in Bengali language applications. Whereas the paper [15] discusses the challenges of addressing hate speech on social media, highlighting its amplification due to online anonymity. It acknowledges the balance between free speech and the harm caused by hate speech, citing the Christchurch attack as an example. The role of social networks in spreading hate and cyberbullying is noted, along with the difficulty in detecting hate speech due to algorithmic biases. It suggests using a trained model to accurately detect and categorize hate speech based on specific criteria like racism and religion.

The paper [5] examines the impact of removing stopwords on the retrieval of knowledge from code mixed with social media data, specifically Bengali–English content. While there is extensive research on SA, language type identification, and text generation for languages mixed with code, the removal of stopwords in this context has been largely overlooked. This study compares corpus-based and non-corpus-based methods for removing stopwords, and it concludes that Mean Average Precision (MAP) values are greatly improved by 16% by corpus-based elimination. The authors also identify the optimal stopword list for each language using tuned TF-IDF threshold values, which enhances information retrieval performance.

In the paper [9], the authors discuss the challenges of addressing hate speech on social networks, highlighting its amplification due to online anonymity. It acknowledges the balance between free speech and the harm caused by hate speech, citing the Christchurch attack as an example. The difficulty of detecting hate speech due to algorithmic biases is noted in light of social media's role in spreading hate and cyberbullying. It suggests using a trained model to accurately detect and categorize hate speech based on specific criteria like racism and religion. The paper [1] reviews recent advancements in SA using DL. The authors discuss various models, including CNNs, RNNs, LSTM networks, and transformer-based models such as BERT and GPT. They cover such issues as model architectures, training strategies, transfer learning techniques, and performance evaluation metrics. The paper presents valuable ideas for researchers and industry professionals interested in leveraging DL for SA applications. Different DL methods were discussed in paper [18].

A method called "Unknown Word Replacement" is proposed in [12]. Unfamiliar terms are substituted with similar familiar words using contextual knowledge. The approach includes pre-processing of data, an algorithm to replace unfamiliar words, and metrics for evaluation. The project intends to increase the parser's accuracy and may lead to significant advances in NLP. The paper [6] is an extensive investigation of multimodal SA. It covers SA across multiple modalities like text, photos, sound, and video. The authors review the existing literature on various approaches, techniques, and trends in multimodal SA, including fusion techniques and DL architectures. They also highlight the challenges and limitations associated with multimodal SA, such as data scarcity, model interpretability, and cross-modal alignment. The survey concludes with information on future research directions and potential avenues for addressing the identified challenges.

3 METHODOLOGY

The entire operations from data preprocessing to result analysis are depicted in Figure 1, which demonstrates our process architecture. The NLTK Python Toolkit is used in all of our experiments.

3.1 Dataset Overview

We collect 7460 comments or lines from various Bengali speakers randomly. These comments are commonly used when communicating and range in length from 3 to 30 Bengali words. People use

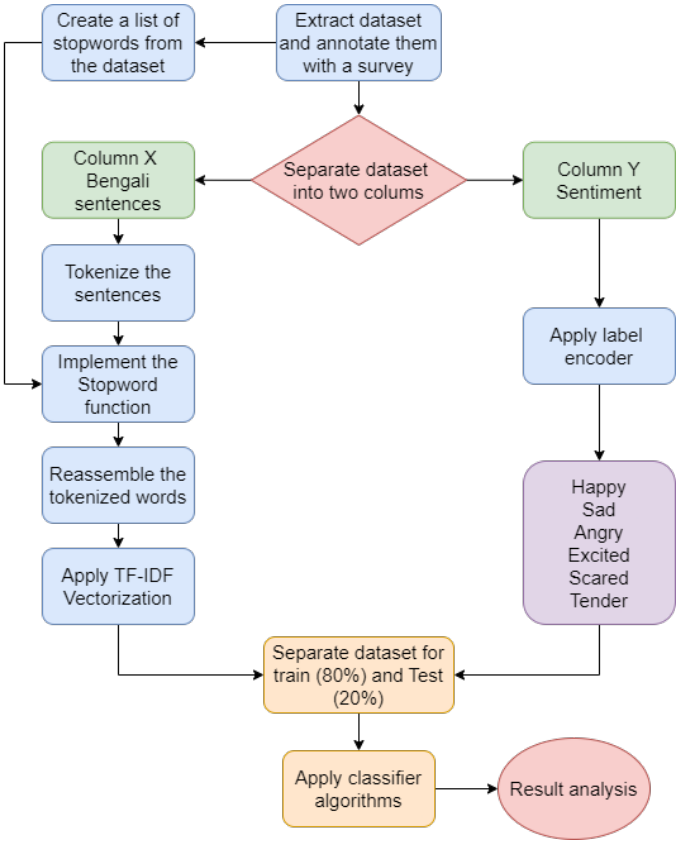

Figure 1: Working Procedure Flow Chart

	A	B
1	**Sentence**	**Sentiment**
2	তাকে সবাই পছন্দ করে	happy
3	আমার কিছু লাগবে না	sad
4	ভয় নেই আমি আছি	tender
5	আমি ভাল আছি	happy
6	মানুষ সব করতে পারে না কেনো	angry
7	আমি যেতে চাই	excited
8	তাকে আমি ভূতের মতো ভয় পাই	scared
9	আনন্দের খবর নেই	sad
10	আমি কেনো কাজ করতে পারি না	sad
11	আমার তাকে পছন্দ না	angry
12	তিনি কোনো কাজ জানেন না	angry
13	তার মত দুষ্ট মানুষ কে সবাই ভয় পায়	scared
14	সে পরীক্ষায় ভালো করেছে	tender
15	সে অজানা আনন্দে বদলে গেছে	excited
16	কেউ নাই ভয় করছে	scared
17	ব্যস্ত আছি বিরক্ত করবে না	angry
18	তার সাহায্য চাইতে আমার ভয় হচ্ছে	scared
19	গরিব লোকদের অনেক কষ্ট	sad
20	তুমি অনেক বেশি নিষ্ঠুর	angry
21	তার চোখে অনেক স্বপ্ন	excited

Figure 2: Sample Dataset

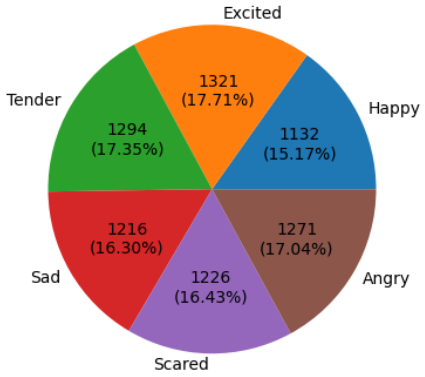

Figure 3: Significance of Each Sentiment

Bengali to express their feelings and emotions about news, incidents, or occurrences. We noticed that 5% -10% of the time, they use the English language to communicate. They also use the English alphabet to write Bengali sentences. We did not include these data in our dataset. Additionally, we excluded data that contained only emotions and no other words or texts. Once we collected all Bengali comments, they were labelled into six categories of sentiments: happy, excited, tender, sad, scared, and angry. Since SA is subjective, annotators used a collaborative voting system to assign each comment to the sentiment category that they perceived to be most appropriate. Five annotators voted on each sentence in the above six sentiment categories, where they had to choose one type of sentiment for each sentence. The sentiment with the majority of votes was selected. In cases where the votes with a majority resulted in a tie with more than one sentiment, the sentence was disregarded. Figure 2 shows a sample of our dataset with six types of classes, which are happy, excited, tender, sad, scared, and angry. Figure 3 shows the number of occurrences and their percentage of six types of classes that exist in our dataset.

3.2 Learning and Testing Model

The K-fold method is a widely used technique in ML that involves training and testing a dataset. In this approach, a specific percentage of the data is used to train the machine to recognize six different classes, while the remaining data is reserved for testing purposes. We chose to allocate 20% of the data for testing, which is 1492 rows, and the remaining 80% of the data, or 5968 rows, for training the model.

During the training process, we applied various standard supervised ML algorithms to train our models. We also used multilabel performance to achieve better results. To find the best fit for our dataset, we experimented with different ML algorithms. After testing, we chose the most suitable algorithms that provided the highest level of accuracy for the final deployment.

3.3 Preprocessing and Feature Extraction

We applied traditional preprocessing techniques to our dataset. Firstly, we tokenized each Bangla term in our dataset. Then, we developed and implemented a function named Stopword that eliminated stopwords from each of the Bengali sentences. Finally, we

recombined the tokenized words into an average sentence. We then constructed a feature matrix where a vocabulary vector represented each analysis. We employed the term frequency-inverse document frequency (TF-IDF) technique to identify the characteristics. We used Label Encoder to classify the dataset by converting classes into numerical values. The presence of messy data, such as abbreviations and unusual forms, can impact sentiment classification in any dataset. Removing stopwords with pre-compiled stopword lists is a commonly used method for reducing noise in textual data. However, in recent years, SA research has made significant progress, and it has been debated if removing stopwords is effective [24].

Our work is related to SA of the Bengali language text dataset. The language in this dataset includes many unnecessary and irregular words. To address this, we created a stopword list based on our dataset and applied it. Mostly we choose Articles, Prepositions, Conjunctions, Pronouns, and Auxiliary Verbs. Below is the list of stopwords we used.
('অনেকে', 'অনেকেই', 'অথবা', 'অথচ', 'সবাই', 'অন্য', 'আজ', 'আপনার', 'আপনি', 'আবার', 'আমরা', 'আমাকে', 'আমাদের', 'আমার', 'আমি', 'আর', 'আগামী', 'অবধি', 'তারে', 'আদ্যভাগে', 'এই', 'একই', 'একে', 'একটি', 'এটি', 'এটা', 'এটাই', 'এবং', 'একবার', 'এবার', 'এদের', 'এঁদের', 'এমন', 'এমনকী', 'এর', 'এরা', 'এঁরা', 'এত', 'এতে', 'এসে', 'একে', 'এ', 'ঐ', 'ইহা', 'ইত্যাদি', 'উনি', 'উপর', 'উপরে', 'ও', 'ওই', 'ওর', 'ওরা', 'ওঁর', 'ওঁরা', 'ওকে', 'ওদের', 'ওঁদের', 'ওখানে', 'কত', 'কবে', 'সেখানে', 'সে', 'স্বয়ং', 'কি', 'কী', 'তিনি', 'তিনিই', 'তিনিও', 'তাঁদের', 'তাঁহারা', 'তাঁরা', 'তাঁর', 'তাঁকে', 'তাই', 'তাকে', 'তাহার', 'তাদের', 'তারা', 'তাকেও', 'তার', 'কারো', 'তুমি', 'তোমার', 'তথা', 'দু', 'দুটি', 'দুটো', 'নিজে', 'নিজেই', 'নিজের', 'নিজেদের', 'প্রভৃতি', 'বার', 'বা', 'ভাবে', 'ভাবেই', 'মধ্যভাগে', 'যাকে', 'যার', 'যারা', 'যাঁর', 'যাঁরা', 'যাদের', 'যিনি', 'যে', 'সবার', 'সহ', 'সুতরাং', 'সহিত', 'সেই', 'সেটা', 'সেটি', 'সেটাই', 'সেটাও', 'সম্প্রতি', 'সেখান', 'হিসাবে', 'জন', 'জনকে', 'জনের', 'জন্য', 'তুলে', 'মোট', 'টি')

Here are a few examples of our dataset, before removing stopwords.

- তাকে সবাই পছন্দ করে
- আমি ভাল আছি
- গরিব লোকদের অনেক কষ্ট

During this phase, each phrase is divided into words or smaller units to separate the content into tokens.

- [তাকে, সবাই, পছন্দ, করে]
- [আমি, ভাল, আছি]
- [গরিব, লোকদের, অনেক, কষ্ট]

After removing the stopwords, the dataset is more efficient and concise. The words from the list of stopwords have been eliminated, resulting in a shorter set.

- [পছন্দ, করে]
- [ভাল, আছি]
- [গরিব, লোকদের, কষ্ট]

After removing the stopwords and putting the tokenized words back together, we get the following sentences.

- পছন্দ করে
- ভাল আছি
- গরিব লোকদের কষ্ট

3.4 Data Analyzing and Implementation of Algorithms

Our proposed framework starts by taking complete Bengali sentences as input. These sentences are then split into individual words to enable further processing. Then, a Stopword list is prepared based on the dataset to filter out common and insignificant words that don't contribute much to SA. The identified words are then removed using the Stopword functionality on the tokenized data. After this preprocessing step, the tokenized data is reassembled to form coherent sentences once again.

Following the preprocessing stage, the data is split into training, testing, and cross-validation sets. This split ensures that the model's performance is evaluated on unseen data, leading to a robust assessment. To enable ML algorithms to process the textual data effectively, and to convert the data into numerical values, we use TF-IDF vectorization [10] techniques and Label Encoder. Finally, the data is partitioned into the specified proportions for training, testing, and cross-validation purposes. We employ various classification approaches and evaluate their performances based on metrics such as accuracy and other parameters, providing insights into the effectiveness of our proposed methodology [2]. Algorithm 1 provides an idea of our proposed framework.

Algorithm 1 Steps for Proposed Framework

Input: Complete Bengali sentences
Output: Sentiment of the input sentences
1: Convert the input sentences section into tokens
2: Prepare a Stopword list according to the dataset
3: Implement Stopword functionality on the tokenized data
4: Reassemble the tokenized data
5: On the training, testing, and cross-validation sets, use Label Encoder and TF-IDF vectorization
6: Splitting them into a train (80%), test (20%), and cross-validation set
7: Apply different classification approaches and demonstrate them based on accuracy and other parameters

3.5 Model Specification and Preprocessing Techniques

We propose a working process that focuses on preprocessing the dataset. Specifically, we use the Bengali dataset and remove the Stopwords to simplify and optimize the training dataset. To achieve this, we employ the Python toolkit's built-in classifiers, including the Neural Network, Random Forest, Decision Tree, Nearest Centroid, Support Vector Machine, and Naïve Bias algorithms [4]. From the literature review, we got the idea to use these models. Most of the papers I cite used these models and got satisfactory outcomes. Additionally, numerous ML tools and Model Specification and Preprocessing Techniques were used, as discussed below.

3.5.1 Neural Network. A neural network is made up of linked layers of neurons that analyze input data and produce output. Biological neurons served as the model for these networks. They learn from data through a process called training, where the network modifies its internal parameters based on the input-output pairings

Table 1: Parameters Setup for CNN Model

Parameters	Value
Batch Size	16
Learning Rate	0.00001
Optimizer	Stochastic Gradient Descent
Loss Function	Categorical Cross Entropy

it observes. This allows the network to develop an understanding of the data it's processing. CNN apply filter layers to local features. Initially developed for computer vision, CNN models have achieved remarkable success and effectiveness in natural language processing (NLP) [11]. CNNs are DL models designed for structured grid-like data like images. They use learned features to identify patterns. They have convolutional, pooling, and fully connected layers. They have performed well in various popular NLP tasks such as semantic parsing, search query processing, recognition sequences, and gait parsing [26]. The following 4 equations illustrate the basic mathematical representation of a convolutional neural network.

- Convolution Operation:

$$z^{[l]}_{\{i,j,k\}} = \sum_{m=0}^{f-1} \sum_{n=0}^{f-1} \sum_{l'=0}^{n_{l-1}} W^{[l]}_{\{m,n,l'\}} \cdot a^{[l-1]}_{\{s+i,t+j,l'\}} + b^{[l]}_{k}$$

- Activation Function (ReLU):

$$a^{\{[l]\}} = \max\left(0,\ z^{\{[l]\}}\right)$$

- Pooling Operation (Max Pooling):

$$a^{[l]}_{i,j,k} = \max_{m=0}^{f} \max_{n=0}^{f} a^{[l-1]}_{s+f\times i+m,t+f\times j+n,k}$$

- Fully Connected Layer:

$$Z^{[l]} = W^{[l]} A^{[l-1]} + b^{[l]}$$

The parameters Setup for CNN is shown in Table 1.

3.5.2 Decision Tree Classifier. Figure The Decision Tree classifier is a statistical model utilized in data science for classification tasks. It is represented graphically as a hierarchical structure with a root node, branches and a leaf node. The primary challenge lies in attribute selection, which determines the feature to split at each level. We opted for this classifier due to its simplicity and transparency, as stated in [16]. The Decision Tree algorithm uses a criterion to determine the best split at each node. One commonly used criterion is the Gini impurity, which measures the impurity of a set of examples. The Gini impurity $Gini(D)$ for a dataset D with N classes is computed in the manner described below:

$$Gini\,(D) = 1 - \sum_{\{i=1\}}^{N} (p_i)^2$$

Where:

- p_i is the probability of an occurrence being classified into the i-th class.
- N is number of classes in total.

This equation represents the Gini impurity criterion, which is used in the Decision Tree algorithm to evaluate the purity of a dataset and determine the best split.

3.5.3 Random Forest. Random Forest is an ML technique that employs multiple decision trees to achieve accurate predictions. The technique builds a large number of decision trees during training to determine the mean forecast for regression or the mode of the classes for classification. Each tree in the prediction is created using a random selection of features from the dataset, and each tree "votes" for the most popular class or provides a prediction. This approach of combining many trees typically performs better than individual decision trees because it reduces overfitting and enhances robustness. Random Forest is particularly useful in dealing with high-dimensional datasets with complex interactions between features, being used as a good choice for various ML tasks. The algorithm can efficiently deal with missing and unbalanced data with a fast runtime, making it an important tool for data scientists and ML practitioners [19]. Neural networks, inspired by biological neurons, adapt through learning, offering a dynamic approach to ML [3]. Decision Tree's key challenge is choosing the root node attribute at each level, typically using information gain or the Gini index for selection [14].

The mathematical formulation of a Random Forest model encapsulates the aggregation of numerous decision trees. Let us define T as the aggregate count of decision trees within the forest. We can represent each decision tree as a function $f_t(x)$, wherein t denotes the tree's index and x signifies the input features. The overall prediction yielded by the Random Forest ensemble is derived from aggregating the predictions of all individual trees:

$$\hat{y}\,(x) = \frac{1}{T} \sum_{t=1}^{T} f_t\,(x)$$

In this equation:

- $\hat{y}(x)$ symbolizes the anticipated result for the input x.
- $f_t(x)$ symbolizes the t-th decision tree's forecast for the input x.
- T is how many decision trees there are in the Random Forest ensemble overall.

3.5.4 Nearest Centroid. The nearest centroid algorithm, also known as the nearest prototype or nearest mean classifier, is a simple and intuitive classification technique used in ML. The centroid of each class's data points in the feature space serves as a representation of that class in this approach. The algorithm calculates its distance to each class centroid and assigns it to the class whose centroid is nearest to the data point. Nearest centroid classification is computationally efficient and works well for datasets with distinct class clusters. Still, it may not perform as effectively in cases where classes overlap or have complex boundaries [4].

The mathematical representation of the nearest centroid algorithm can be expressed as follows:

$$\text{Class (x)} \;=\; \arg\min_{\text{i}} x - \mu_i$$

Here, Class(x) denotes the class assignment for the data point x, μ_i represents the centroid vector of class i, and denotes the Euclidean distance. The algorithm iterates over all classes to find the one with the closest centroid to the given data point x, effectively assigning x to that class. This approach simplifies classification tasks by considering the distance between data points and class centroids, making it a straightforward yet effective method for certain types of datasets.

3.5.5 Support Vector Machine. SVM is a robust supervised learning algorithm that is used for classification also regression tasks. Finding the optimal hyperplane is how it operates and that separates different classes in a feature space, to maximize the margin. The most effective hyperplane to divide the categories is found, and it should have good generalization to new data. SVM aims to optimize the margin in classification work while guaranteeing accurate categorization of all data points [4].

The mathematical formulation of a Soft Margin Linear SVM can be articulated as an optimization challenge. Given a training dataset $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, with x_i denoting the input features and y_i signifying the corresponding class labels, the objective of SVM is to identify the optimal hyperplane defined by $w^T x + b = 0$. This hyperplane should not only maximize the margin but also ensure accurate classification of the training data. This objective leads to a specific constrained optimization problem formulation.

$$\frac{1}{2}|w|^2 + C\sum_{i=1}^{n} \xi_i$$

$$y_i \left(w^T x_i + b\right) \geq 1 - \xi_i, \quad i = 1, 2, \ldots, n$$

$$\xi_i \geq 0, \quad i = 1, 2, \ldots, n$$

Where:

- w represents the weight vector orthogonal to the hyperplane,
- b denotes the bias term,
- C is a parameter that controls how much margin is maximized and how much mistake is minimized,
- ξ_i show the distance of data points from the margin or the classification error as slack variables,
- $||w||^2$ symbolizes the weight vector's squared norm w,
- (x_i, y_i) are the training data points
- y_i are the respective class labels.

The goal is to minimise the norm of the weight vector while ensuring that all data points are accurately classified with a minimum margin of $1 - \xi_i$. The parameter C governs the penalty for misclassification or margin violations, where higher values of C result in a narrower margin but can improve classification accuracy.

3.5.6 Naïve Bayes. Naïve Bayes is simple yet effective and assumes feature independence, which makes it efficient for tasks such as text classification and spam filtering. Although this assumption isn't always correct and may lead to inaccuracies, its performance with high-dimensional data keeps it a good choice [4].

The mathematical representation of Naïve Bayes classifier can be expressed as follows:

$$P(y \mid x_1, x_2, \ldots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, x_2, \ldots, x_n)}$$

Where:

- $P(x_1, x_2, \ldots, x_n)$ is the y class's posterior probability for features $x_1, x_2, ..., x_n$.
- $P(y)$ is the y class's prior probability.
- $P(x_i \mid y)$ is the probability conditional xi on the feature given class y.
- $P(x_1, x_2, \ldots, x_n)$ is the probability of identifying characteristics $x_1, x_2, ..., x_n$.

In the framework of Naïve Bayes, the term "naïve" refers to the presumption that the features $x_1, x_2, ..., x_n$ are independent under conditions given the class y. This assumption allows for the simplification of the expression:

$$P(y \mid x_1, x_2, \ldots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, x_2, \ldots, x_n)}$$

This expression is used in Naïve Bayes to predict the class with the highest posterior probability based on given features.

3.5.7 K-fold Cross-validation Method. This is a process of evaluating a model's effectiveness that entails splitting a dataset into k subsets, with each iteration using the remaining subsets for training and one subset serving as a test set. The procedure is carried out k times, and the model's efficacy is estimated based on the average performance throughout all iterations. The following equation is the mathematical representation for the K-fold Cross-validation Method [27].

$$\text{Performance} = \frac{1}{k} \sum_{i=1}^{k} \text{Metric}_i$$

Where:

- Metric_i : represents the performance metric obtained on the i-th fold during testing.
- k is the number of folds.

3.5.8 Tokenization. Tokenization is a foundational system where the text is divided into smaller units, called tokens. These units can include words, phrases, symbols, or other significant elements, varying according to the task's context and specific requirements. Tokenization is an important first step in text preparation in NLP that helps make important tasks like named entity identification, part-of-speech tagging, and SA easier to do [23].

3.5.9 TF-IDF Vectorization. The TF-IDF Vectorization is a technique in NLP that is used to transform text into numerical vectors. It calculates a word's relevance for a corpus by multiplying term frequency (TF) and inverse document frequency (IDF). Emphasize terms that are significant and unique to a given document. The equation here is the formula for the TF-IDF vectorization [2].

$$\text{TF} - \text{IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D)$$

Where:

- t denotes a term (word) in a document.
- d signifies a specific document.
- D stands for the entire corpus of documents.
- TF(t, d) represents the term frequency of t in d, measuring how frequently t appears in d.
- IDF(t, D) represents the opposite document frequency of t in D, assessing the rarity of t across all documents in D.

3.5.10 Stopword Removal. Stopwords are words in a language that are commonly used and are often excluded during text data preprocessing in NLP tasks. Words such as "the", "and", "is", "in", etc. are considered less important and are eliminated. This exclusion of stopwords helps to reduce the size of the dataset, making analysis more efficient [23].

Table 2: Performance Comparison Before Implementing Stopword Function

Model	Accuracy	F1 Score	Recall	Precision
Neural Network	72.87	73.19	71.97	72.14
Random Forest	71.71	72.71	71.71	72.64
Decision Tree	71.60	72.57	71.60	72.54
Nearest Centroid	70.71	70.71	70.85	70.77
SVM	69.30	69.90	68.33	69.28
Naïve Bias	68.66	67.66	68.58	67.65

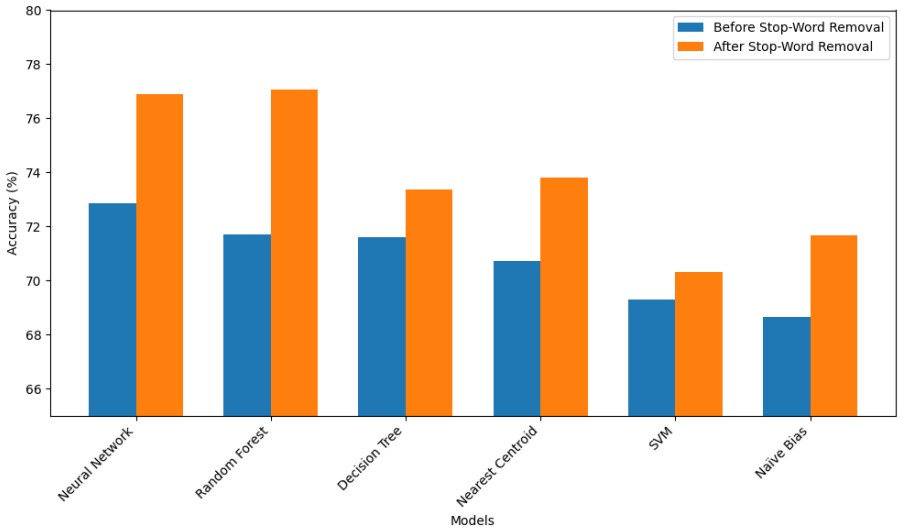

Figure 4: Comparison of Matrices

3.5.11 Label Encoder. Label encoder is a crucial tool in ML that converts categorical labels into numerical representations. Before training an ML model, it is necessary to transform categorical data into numerical form. Label Encoder is a helpful tool in this regard. It assigns a unique integer value to each category and label to encode them into numerical form. This conversion allows ML algorithms to efficiently handle categorical data because mathematical processes are better suited for numerical data. Label encoding is a popular preprocessing step for various ML tasks, such as clustering, regression, and classification [23].

4 RESULT ANALYSIS

The whole working procedure is illustrated through a flow chart in Figure 1. In this section, we have collected the evaluation and analysis results of all the considered methods. Several performance measures have been used to examine and assess these techniques. The goal of all these performance evaluation criteria is to prevent overfitting and determine which strategy out of all the ones utilized in this work is the best. The methods' comparative analyses are displayed below.

Our proposed Bangla dataset for testing purposes was performed on the trained model after the training had been completed. Below table 2 and 3 along with Figure 4 demonstrate the overall outcome.

Table 3: Performance Comparison After Implementing Stopword Function

Model	Accuracy	F1 Score	Recall	Precision
Neural Network	76.91	76.64	76.91	75.50
Random Forest	77.05	76.64	77.05	77.16
Decision Tree	73.37	75.36	75.37	75.50
Nearest Centroid	73.81	74.21	73.85	74.17
SVM	70.30	72.90	73.33	73.28
Naïve Bias	71.66	72.66	72.58	72.65

From Table 2 and Table 3, it is clearly shown that, after the removal of stopwords from the dataset Accuracy, F1 Score, and recall score, the Precision score is improved for all six algorithms. Almost 2% to 6% of accuracy increases after the removal of stopwords. In all parts before and after using stopword, the Random Forest Classifier obtained the highest accuracy. Since Random Forests are made up of several single trees, each dependent on a random sample of training data, they are difficult to understand. As the Random Forest tree is fully grown and unpruned, it divides the feature space into smaller regions.

We mostly focus on the improvement of accuracy after using the Stopword function in our work. Below, a graph in Figure 4 describes only accuracy differences.

The tables illustrate a comparison of the performance of SA models before and after the implementation of the Stopword function. The models achieved moderate accuracy before implementing stopword removal, with the Neural Network leading at 72.87%, closely followed by Random Forest and Decision Tree. However, after stopword removal, there was a significant improvement in performance across all models. The Neural Network, Random Forest, and Decision Tree models demonstrated significant accuracy gains, with the Neural Network achieving the highest accuracy of 76.91% after Random Forest which is 77.05%. This suggests that stopword removal effectively enhanced the models' ability to discern sentiment from text data. Additionally, the F1 score, recall, and precision metrics also improved post-stopword removal, indicating better overall model performance in correctly identifying sentiment. Notably, the SVM model exhibited a slight decrease in accuracy after stopword removal, suggesting that its performance may not have been as influenced by the presence of stopwords compared to other models. Overall, the results underscore the importance of preprocessing techniques like stopword removal in enhancing SA model performance.

5 CONCLUSION AND FUTURE WORK

Due to the language's diversity, variation, and complex grammar, SA in Bengali poses unique challenges. Despite these challenges, our study reveals significant opportunities for developing innovative solutions for analyzing sentiment in Bengali text and speech. The result shows preprocessing techniques, such as stopword removal can significantly improve the performance of SA systems We observed noticeable improvements across various ML models by creating a customized list of Bengali stopwords and implementing the Stopword function. The Neural Network and Random Forest classifiers showed the most significant accuracy improvements, with the Random Forest model achieving the best precision of 77.05% after stopword removal. These results highlight the effectiveness of preprocessing techniques in refining SA tasks, particularly in a linguistically rich and diverse language like Bengali.

Future work should focus on refining these models and exploring additional preprocessing techniques to improve performance. This expansion would create a more comprehensive model capable of handling diverse linguistic contexts. Given the observation that Bengali speakers often mix English with Bengali, future work could also explore models that can handle code-switching between Bengali and English. This kind of text would involve developing algorithms capable of efficiently processing and analyzing multilingual text. Additionally, investigating advanced preprocessing techniques, such as stemming and lemmatization, alongside stopword removal can upgrade the model's ability to identify the root forms of words and improve sentiment classification accuracy.

While this study utilized traditional ML algorithms, future work could explore DL models for SA. These models have shown great promise in handling large and complex datasets. Developing a real-time SA system for Bengali could have practical applications in customer feedback analysis, social media monitoring, and public opinion tracking, necessitating efficient and scalable models capable of processing real-time data streams. Additionally, creating user-friendly interfaces for SA tools could make these models accessible to a broader audience.

Acknowledgments

This work partially utilized AI-based tools, like Grammarly, to refine language and structure content. The authors reviewed and verified all AI-generated material for accuracy and compliance with ethical standards.

References

[1] Tariq Abdullah and Ahmed Ahmet. 2022. Deep learning in sentiment analysis: Recent architectures. Comput. Surveys 55, 8 (2022), 1–37.
[2] Haisal Dauda Abubakar, Mahmood Umar, and Muhammad Abdullahi Bakale. 2022. Sentiment classification: Review of text vectorization methods: Bag of words, Tf-Idf, Word2vec and Doc2vec. SLU Journal of Science and Technology 4, 1 & 2 (2022), 27–33.
[3] Hana H Alalawi and Manal S Alsuwat. 2021. Detection of cardiovascular disease using machine learning classification models. International Journal of Engineering Research & Technology 10, 7 (2021), 151–7.
[4] Ethem Alpaydin. 2020. Introduction to machine learning. MIT press.
[5] Supriya Chanda and Sukomal Pal. 2023. The effect of stopword removal on information retrieval for code-mixed data obtained via social media. SN Computer Science 4, 5 (2023), 494.
[6] Ringki Das and Thoudam Doren Singh. 2023. Multimodal sentiment analysis: a survey of methods, trends, and challenges. Comput. Surveys 55, 13s (2023), 1–38.
[7] Zannatul Ferdous, Rabeya Akhter, Anika Tahsin, Sumaiya Nuha Mustafina, and Nuzhat Tabassum. 2022. Sentiment Analysis on COVID-19 Vaccine Twitter Data using Neural Network Models. In Proceedings of the 2nd International Conference on Computing Advancements. 435–441.
[8] Md Muzakker Hossain, Zahin Awosaf, Md Salman Hossan Prottoy, Abu Saleh Muhammod Alvy, and Md Kishor Morol. 2022. Approaches for improving the performance of fake news detection in bangla: Imbalance handling and model stacking. In Proceedings of International Conference on Fourth Industrial Revolution and Beyond 2021. Springer, 723–734.
[9] Muhammad Okky Ibrohim, Cristina Bosco, and Valerio Basile. 2023. Sentiment analysis for the natural environment: A systematic review. Comput. Surveys 56, 4 (2023), 1–37.
[10] Zeyi Jin, Xin Lai, and Jingjig Cao. 2020. Multi-label sentiment analysis base on BERT with modified TF-IDF. In 2020 IEEE International Symposium on Product Compliance Engineering-Asia (ISPCE-CN). IEEE, 1–6.
[11] Yue Kang, Zhao Cai, Chee-Wee Tan, Qian Huang, and Hefu Liu. 2020. Natural language processing (NLP) in management research: A literature review. Journal of Management Analytics 7, 2 (2020), 139–172.
[12] Raihan Kibria and Khandaker Tabin Hasan. 2020. Improving Natural Language Parser Accuracy by Unknown Word Replacement. In Proceedings of the International Conference on Computing Advancements. 1–7.
[13] Yuki Kuwabara and Yu Suzuki. 2023. Attention based stopword generation for neural network based text processing. In International Conference on Information Integration and Web Intelligence. Springer, 526–540.
[14] John F Magee. 1964. Decision trees for decision making. Harvard Business Review Brighton, MA, USA.
[15] Tahbib Manzoor, Md Wahidur Rahman Araf, Monjurul Sharker Omi, Tanvir Ahmed Abir, Arpan Das Abir, Irin Hoque Orchi, and Faisal Bin Ashraf. 2023. Mitigating Online Harassment: Machine Learning Approaches for Hate Speech Detection in Transliterated Bengali Comments. In 2023 26th International Conference on Computer and Information Technology (ICCIT). IEEE, 1–6.
[16] Abdul Fattah Mashat, Mohammed M Fouad, S Yu Philip, and Tarek F Gharib. 2012. A decision tree classification model for university admission system. International Journal of Advanced Computer Science and Applications 3, 10 (2012).
[17] G. Mostafa, Ikhtiar Ahmed, and Masum Shah Junayed. 2021. Investigation of different machine learning algorithms to determine human sentiment using Twitter data. International Journal of Information Technology and Computer Science (IJITCS) 13, 2 (2021), 38–48.
[18] G Mostafa, Md Shifat Hamidi, and Dewan Md Farid. 2023. Detecting Lung Cancer with Federated and Transfer Learning. In 2023 26th International Conference on Computer and Information Technology (ICCIT). IEEE, 1–6.
[19] Karandi Nurbagja, Nurirwan Saputra, Ahmad Riyadi, Meilany Nonsi Tentua, *et al.* 2023. Sentiment Analysis of the Increase in Fuel Prices Using Random Forest Classifier Method. Buletin Ilmiah Sarjana Teknik Elektro 5, 1 (2023), 132–144.
[20] SM Samiul Salehin, Rasel Miah, and Md Saiful Islam. 2020. A comparative sentiment analysis on Bengali Facebook posts. In Proceedings of the international

conference on computing advancements. 1–8.
[21] Kogilavani Shanmugavadivel, VE Sathishkumar, Sandhiya Raja, T Bheema Lingaiah, S Neelakandan, and Malliga Subramanian. 2022. Deep learning based sentiment analysis and offensive language identification on multilingual codemixed data. Scientific Reports 12, 1 (2022), 21557.
[22] Riffat Sharmin, Shantanu Kumar Rahut, and Mohammad Rezwanul Huq. 2020. Bengali spoken digit classification: A deep learning approach using convolutional neural network. Procedia Computer Science 171 (2020), 1381–1388.
[23] Jannatul Ferdousi Sohana, Ranak Jahan Rupa, and Moqsadur Rahman. 2021. Bengali Stop Word Detection Using Different Machine Learning Algorithms. In Proceedings of International Joint Conference on Advances in Computational Intelligence: IJCACI 2020. Springer, 131–142.
[24] Jie Tao and Xing Fang. 2020. Toward multi-label sentiment analysis: a transfer learning based approach. Journal of Big Data 7, 1 (2020), 1.
[25] Ramesh Vatambeti, Srihari Varma Mantena, KVD Kiran, M Manohar, and Chinthakunta Manjunath. 2024. Twitter sentiment analysis on online food services based on elephant herd optimization with hybrid deep learning technique. Cluster Computing 27, 1 (2024), 655–671.
[26] Zengbin Wang, Saihui Hou, Man Zhang, Xu Liu, Chunshui Cao, and Yongzhen Huang. 2023. GaitParsing: Human Semantic Parsing for Gait Recognition. IEEE Transactions on Multimedia (2023).
[27] Xinyu Zhang and Chu-An Liu. 2023. Model averaging prediction by K-fold cross-validation. Journal of Econometrics 235, 1 (2023), 280–301.