A Study on usage of various Ensemble Learning Models for Sentiment Analysis

Krishna Sai Kunche*
CSE Department
VFSTR Deemed to be University
Guntur, India
krishna.11.kunche@gmail.com

Yaswanth Suryadevara†
CSE Department
VFSTR Deemed to be University
Guntur, India
yaswanthsuryadevara@gmail.com

Gopi Chandu Pamidi‡
CSE Department
VFSTR Deemed to be University University
Guntur, India
gopichanduchandu652@gmail.com

Sri Sai Divyakola§
CSE Department
VFSTR Deemed to be University
Guntur, India
srisaidivyakola@gmail.com

Dr. Venkatrama Phani Kumar S§
Professor, Department of CSE
VFSTR Deemed to be University
Guntur, India
svrphanikumar@yahoo.com

Dr. Venkata Krishna Kishore K§
Professor
Head, Department of CSE
VFSTR Deemed to be University
Guntur, India
kishorekvk1@*yahoo.com*

***Abstract*—This paper investigates sentiment analysis on social media platforms, with a primary focus on Twitter posts. Understanding user emotions towards social media content is crucial for various applications, particularly marketing. The study employs ensemble learning techniques, with a specific focus on text data analysis. Feature extraction methods, including TF-IDF, count vectorization, Word2Vec[1], and GloVe, are utilized alongside ensemble learning classifiers. Initially, data is sourced from Kaggle, and to address dataset imbalance, SMOTE (Synthetic Minority Over-sampling Technique) is applied. Preprocessing steps involve NLTK for removing stopwords, tokenization, and URL elimination. Sentiment analysis is then conducted using classifiers such as Random Forest, CatBoost, XGBoost, and Stacking. Among these, Random Forest emerges as the most accurate classifier for sentiment analysis. This research aims to improve sentiment analysis accuracy through the integrating NLTK, Feature extraction and Ensemble learning techniques.**

Keywords: Tweet Sentiment, CatBoost, Stacking, XGBoost, Random Forest, NLTK.

I. Introduction

With the rapid expansion of social media platforms, the opinions and sentiments of individuals are increasingly shaped by the content they encounter online. Among these platforms, Twitter stands out as a superior source for understanding the views and sentiments of the public. As users engage with a diverse array of topics, ranging from current events to personal anecdotes, Twitter serves as a rich source of data to know the opinion of the public.The capability to comprehend and assessing sentiment expressed in tweets holds significant value across various domains such as politics,marketing, and opinion monitoring. By evaluating the sentiments of Twitter users regarding particular topics or events, stakeholders can acquire invaluable insights into public perception, enabling them to adapt and refine their strategies effectively.

This research aims to explore sentiment analysis on tweets, focusing on the detection and interpretation of opinions expressed by users. Leveraging ensemble learning and natural language processing techniques, the study seeks to uncover patterns in tweet data that reveal prevailing sentiments. By investigating methodologies for sentiment analysis, including feature extraction and Ensemble learning classification, the aim is to enhance the accuracy and effectiveness of sentiment interpretation.

This paper demonstrates sentiment analysis in Twitter captions using different algorithms. It begins by categorizing the captions into two polarities: positive (1) and negative (0). The data is trained , and sentiment is predicted using the Random Forest [14], CatBoost, XGBoost[7], and Stacking algorithms.

- This research advances sentiment analysis on social media, particularly Twitter, through methodologies aimed at enhancing accuracy and effectiveness.
- Leveraging Ensemble techniques and natural language processing, the study explores various feature extraction methods and Ensemble learning classification algorithms.
- The goal is to uncover patterns in tweet data that reflect prevailing sentiments.
- The findings provide valuable insights for stakeholders in domains such as politics , marketing analysis.
- These insights enable stakeholders to gain a deeper understanding of public perception and tailor their strategies accordingly.
- By improving the accuracy of sentiment analysis on Twitter, the study offers practical implications for real-world applications.

II. RELATED WORK

The researchers employed Twitter as a platform to monitor and anticipate depression symptoms among individuals. To delineate depressive tendencies, they utilized crowd-sourcing and introduced various social media markers, including emotions, language patterns, and user engagement.

Le and Nguyen utilized classifiers, specifically Naive Bayes and Support Vector Machine , to categorize sentiment as positive or negative. Additionally, they employed a robust feature extractor to enhance accuracy in sentiment analysis on Twitter data [8]. Gupta et al. perform data collection methods, pre-processing steps, feature extraction techniques, and sentiment classifiers employed for Twitter sentiment analysis. It highlights the use of Python libraries such as NLTK and Scikit-learn to facilitate implementation and experimentation in the field [4]. Gautam and Yadav propose machine learning techniques used for classify sentence and reviews based on tweets data . They aim to analyze labeled reviews from Twitter datasets using techniques like maximum entropy, naive bayes and SVM. Their findings suggest that combining Naive Bayes with a unigram model and semantic analysis using WordNet improves accuracy to 89.9% [3].

Neethu et al. explored sentiment on Twitter using ml techniques. They addressed challenges in identifying emotional keywords, such as misspellings and slang words, by proposing a two-step feature extraction process. This process incorporated both Twitter-specific and general text features into the feature vector. They evaluated classification accuracy using classifiers like SVM, Maximum Entropy, Navie Bayes, and Ensemble classifiers, finding similar accuracy across all [10]. Ramadhani and Goo proposed a sentiment analysis method for tweets involving text cleaning, preprocessing, and deep learning using a neural network architecture. They utilized text mining for information extraction and employed cleaning techniques such as stemming, lowercase conversion, and stopword removal. The DNN has three hidden layers and ReLU, and sigmoid activations. Optimization was done using Mean Square Error and Stochastic Gradient Descent [11]. Yadav et al. studied on sentiment analysis using supervised machine learning . They compared the performance of some Ensemble models. Despite Linear SVC achieving the accuracy of 83.71%, Logistic Regression was chosen for classification due to its balanced performance across accuracy and AUROC [13].

Mandloi and Patel conducted Twitter sentiment analysis using text preprocessing techniques like lowercase conversion, tokenization, and stop word removal. They employed feature extraction methods such as parts of speech tagging and various feature selection techniques. For sentiment analysis, they utilized machine learning classifiers , evaluating their performance based on accuracy and precision metrics [9]. Kolchyna et al. compares different lexicon combinations, discusses feature generation and selection, and evaluates methods using SemEval-2013 Twitter datasets. Results favor machine learning methods over lexicon-based approaches. The paper proposes an ensemble method combining lexicon scores with machine learning for improved accuracy. Additionally, employing a cost-sensitive classifier enhances sentiment classification performance by up to 7% [6]. Sahayak et al. utilized hand-annotated dictionaries for emoticons and acronym dictionaries from the web. Models like Naive Bayes,MaxEnt and SVM. Feature extractraction is done by comprising unigrams, bigrams, and unigrams with tags of parts of speech. [12].

III. METHODOLOGY

A. *Dataset:*

The dataset utilized in this study is sourced from Kaggle and comprises text data, representing tweets and their corresponding targets, indicating whether they are classified as positive or negative sentiments. Specifically, the dataset consists of 31,962 tweets, each labeled with a target value of 1 for positive sentiment and 0 for negative sentiment. It's noteworthy that the dataset exhibits a class imbalance, with 29,600 instances labeled as negative, and the remaining instances labeled as positive. This imbalance poses a challenge that needs to be addressed in the subsequent analysis to ensure robust and accurate sentiment classification.

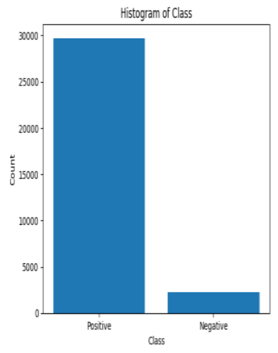

Fig. 1. number of samples in each class

Data is balanced using SMOTE . SMOTE adds some synthetic data points to the minority class to balance the dataset. This helps in addressing class imbalance issues and can lead to an increase in the accuracy score of the proposed machine learning model.

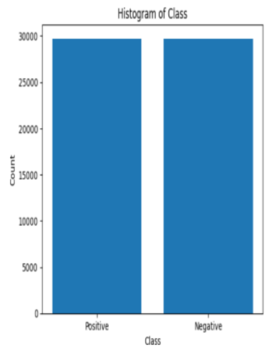

Fig. 2. After balancing the number samples in each class

The data balancing technique SMOTE is applied to address class imbalance, as illustrated in Figure 2. Subsequently, the data undergoes preprocessing using natural language processing (NLP) techniques to extract feature vectors. Ensemble methods are then employed on the processed training data to construct the training model, which is subsequently utilized for data classification. This methodology adheres to the high-level process flow outlined in the accompanying figure.

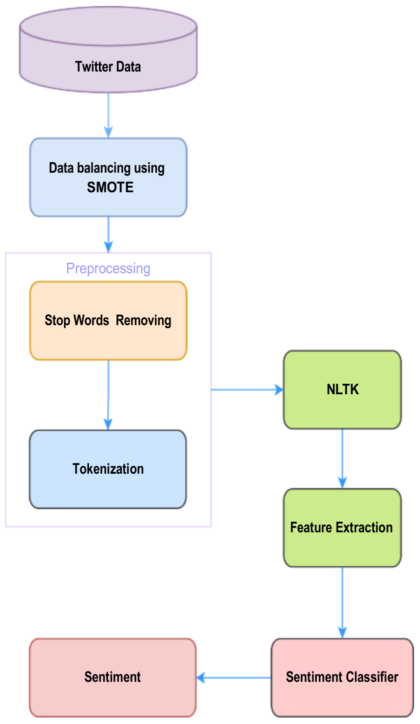

Fig. 3. Generic Architecture of Proposed Methodology

B. preprocessing:

Twitter data often contains significant amounts of noise, including URLs, special characters, and irrelevant words, which can distort sentiment analysis results. Therefore, preprocessing is crucial to clean and standardize the text for accurate analysis. Initially, noise elements such as URLs, special characters, hashtags, and mentions are removed to focus solely on the tweet content. Following this, tokenization is employed tobreak the text into words or tokens, facilitating further analysis.To maintain consistency and reduce complexity, all text is converted to lowercase to mitigate discrepancies arising from case variations. Additionally, common stopwords, which are words with minimal semantic value, are systematically eliminated to reduce data dimensionality and emphasize meaningful words. Furthermore, stemming or lemmatization techniques are applied to standardize word forms and minimize redun- dancy in the text. Through these preprocessing steps, the data is thoroughly cleaned and standardized, providing a solid foundation for accurate sentiment analysis.

C. NLTK:

The Natural Language Toolkit (NLTK) [2] was utilized to process the text data, followed by the evaluation of sentiment scores. Subsequently, the accuracy of sentiment predictions was assessed using the Random Forest, CatBoost, XGBoost, and Stacking algorithms.

NLTK (Natural Language Toolkit) is a potent Python library for NLP, enabling tasks like tokenization, stemming, and sentiment analysis. Its tokenization feature breaks text into words

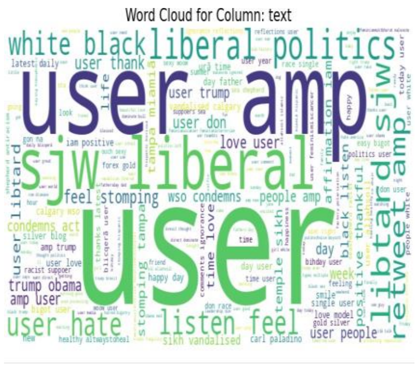

Fig. 4. Words frequency in the twitter dataset Represented in wordcloud

or sentences, aiding analysis. Stemming and lemmatization reduce words to their root forms, simplifying analysis by treating variations of the same word equally. NLTK provides sentiment lexicons, dictionaries associating words with sentiment scores, facilitating sentiment assignment to text.

In this research, NLTK is a highly useful tool for removing unwanted words and accurately identifying words through lemmatization. With NLTK, Twitter data is cleaned and prepared for sentiment analysis.

D. Feature Extraction and Selection :

TF-IDF:One of the best feature extraction methods for text mining is term frequency-inverse document frequency, which provides information on the importance of words in texts. In TF1IDF, ”TF” stands for ”Term Frequency,” indicating how frequently the term appears in the document, and ”IDF” for ”Inverse Document Frequency,” which measures how uncommon the word is throughout the corpus. To be more precise, TF measures how frequently a term appears in a document, whereas IDF assesses how unique a word is over the entire corpus of papers.The TF and IDF are multiplied to get the outcome.

The TF-IDF calculation formula is given by:

$$\text{TF-IDF} = \text{TF} \times \text{IDF}$$

Where:

$$\text{TF} = \frac{d}{\text{number of times term } t \text{ in given document } d}$$

$$\text{IDF} = \log \frac{\text{no. of documents containing term } t + 1}{\text{Total number of documents in corpus}}$$

Document containing the terms with higher TF-IDF score can considers as more relevant.it is best feature extraction technique for the tasks such as sentiment analysis,spam detection etc.

Word2Vec: Indeed, one of the most potent methods in natural language processing (NLP) is word embedding, such as Word2Vec. Word2Vec identifies semantic links and similarities between words by expressing words as dense vectors in a continuous vector space. This helps algorithms better understand the contextual meaning of words in a given language.Word embedding methods such as Word2Vec transform the way natural language data is represented and handled in NLP tasks. They are essential tools for a variety of natural language processing (NLP) applications, including as text classification, sentiment analysis, machine translation, and more, since they allow computers to comprehend the semantic subtleties of language, lessen the impact of dimensionality, and enable transfer learning.

GloVe: Words can be densely vector represented using GloVe (Global Vectors for Word Representation) embeddings, which are based on the co-occurrence statistics of words in huge text corpora. These pre-trained embeddings can be utilized directly as features for sentiment analysis tasks, capturing semantic links between words. Text documents can be represented as fixed-size vectors appropriate for input into machine learning models by mapping words to their corresponding GloVe vectors and aggregating them at the document level (e.g., via averaging). GloVe embeddings are an effective means of utilizing semantic information to enhance the efficacy of sentiment analysis models, particularly in situations with sparsely labeled data or text corpora with a narrow focus.

Count Vectorization: Count Vectorization is a fundamental technique in NLP [5] that converts text documents into numerical representations. Tokenizing the text and counting the frequency of each word in each document are the steps involved. The result is a sparse matrix with each cell reflecting the frequency of the relevant token in the related document, rows representing documents and columns representing unique tokens. Count vectorization is a simple and effective method, but it merely records whether words are present or absent in a document; it ignores word order or semantic significance. However, it provides a numerical representation of text data that can be fed into machine learning algorithms, making it a fundamental step for a number of NLP tasks like text classification, sentiment analysis, and document clustering.

By using these various feature extraction techniques this research gained knowledge that which feature is used for required tasks and beest and effective feature extraction.

E. List of models used for classification:

Random Forest: A reliable supervised learning technique is Random Forest. It builds several decision trees during training and outputs a class that is the mean of the individual trees or the mode of the classes. Random Forest is a well-liked option for numerous machine learning applications becauseof its flexibility.The best classifier for preventing overfitting isthis one.Using a set of decision trees that are frequently taughtvia the "bagging" procedure, it creates a forest. This bagging strategy's main goal is to increase output by mixing different learning models. To provide forecasts that are more precise and reliable, random forests merge several decision trees.

The feature importance FI_{ij} of feature i in tree j can be calculated as follows:

$$\mathrm{FI}_{ij} = \frac{\Delta\mathrm{Impurity}_{ij}}{N}$$

where $\Delta\mathrm{Impurity}_{ij}$ is the total decrease in impurity due to splits on feature i in tree j, and N is the total number of samples in the dataset.

After iterating through all trees, the aggregated importance of feature i across all trees is computed as:

$$\mathrm{RF\_FI}_i = \frac{1}{T}\sum_{j=1}^{T} \mathrm{FI}_{ij}$$

where T is the total number of trees in the Random Forest model.

CatBoost: Renowned for its resilience and efficiency, CatBoost is a potent gradient boosting system. To optimize performance, a great deal of fine-tuning was done in thisstudy by experimenting with different model parameters and configurations. The goal was to make the most of CatBoost's capabilities by using strategies like early halting to reduce overfitting and modifying the learning rate to guarantee ideal convergence.

XG-Boost: XG-Boost is a versatile framework for gradient boosting, known for its high performance and efficiency. In this study,To optimize performance, more fine-tuning was doneby delving deeper into the model's parameters and setups. Thegoal was to fully exploit XGBoost by utilizing strategies like early halting to prevent overfitting and modifying the learningrate to achieve optimal convergence.

In both XGBoost and CatBoost, the final prediction is calculated by :

$$\hat{y}_x = \sum_{i=1}^{T} F_i(x)$$

where $\hat{y}_x$ represents the final prediction for the input x, and $F_i(x)$ denotes the prediction made by the i-th weak learner.

Stacking: In order to improve model performance, this study used stacking, a potent ensemble learning technique. Many base learners and meta-learner configurations were investigated in order to fine-tune the stacking technique through extensive experimentation. The principal objective was to use the combined predictive capacity of several models through the deliberate integration of their outputs. To optimize the stacking ensemble and guarantee resilience and generalization, methods like model blending and cross-validation were applied.
In Stacking the final predection is calculated:

$$\hat{y}_x = \mathrm{MetaModel}(\mathrm{BaseModel}_1(x), \ldots\ldots, \mathrm{BaseModel}_n(x))$$

where $\hat{y}_x$ is the final prediction, and $\mathrm{BaseModel}_i(x)$ are the predictions made by the base models.

IV. Experimental results and discussion:

Sentiment analysis was conducted using various algorithms in a COLAB environment, and the resulting metrics are presented below. The assessment primarily focuses on four key parameters: precision, recall, F1-score, and support. These metrics provide insights into the effectiveness of each algorithm in accurately predicting sentiment from the data.

Performance comparison of various classifiers models with different Feature Extraction Techniques:

Feature extraction techniques were applied independently to each classifier model. Table I provides an overview of the results achieved by employing the TF-IDF method.

TABLE I
Classifiers with TF-IDF

Classifiers	**Accuracy**	**Precision**	**Recall**	**F1-Score**
XGBoost	0.93	0.92	0.97	0.94
CatBoost	0.95	0.94	0.96	0.95
Stacking	0.96	0.95	0.97	0.96
Random Forest	0.99	0.99	0.99	0.99

Table 1 presents the performance metrics of different classifiers using the TF-IDF feature extraction technique. Random Forest stands out with the highest scores across all metrics, including accuracy (99%), precision (99%), recall (99%), and F1-score (99%). This indicates Random Forest's superior ability to accurately classify instances compared to other classifiers in the table.

Feature extraction techniques were applied independentlyto each classifier model. Table II provides an overview ofthe results achieved by employing the Count vectorizationmethod.

TABLE II
Classifiers with Count Vectorization

Classifiers	**Accuracy**	**Precision**	**Recall**	**F1-Score**
XGBoost	0.94	0.92	0.97	0.98
CatBoost	0.96	0.95	0.97	0.96
Stacking	0.97	0.96	0.98	0.97
Random Forest	0.98	0.99	0.99	0.99

Table II compares the performance of different classifiers using the Count Vectorization feature extraction technique. Random Forest achieves the highest scores across all metrics, with 98% accuracy, 99% precision, recall, and F1-Score. This suggests Random Forest's robust performance in classifying instances based on Count Vectorization

Feature extraction techniques were applied independently to each classifier model. Table III provides an overview of the results achieved by employing the Word2Vec method.

Table III presents the performance metrics of different classifiers using the Word2Vec feature extraction technique. Random Forest achieves the highest scores across all metrics, with 98% accuracy, 97% precision, 99% recall, and 98% F1-Score. This indicates Random Forest's strong performancein classifying instances based on Word2Vec embeddings.

Feature extraction techniques were applied independently to each classifier model. Table IV provides an overview of the results achieved by employing the GloVe method.

TABLE III
Classifiers with Word2Vec

Classifiers	**Accuracy**	**Precision**	**Recall**	**F1-Score**
XGBoost	0.95	0.93	0.98	0.96
CatBoost	0.96	0.96	0.98	0.97
Stacking	0.96	0.95	0.97	0.96
Random Forest	0.98	0.97	0.99	0.98

TABLE IV
Classifiers with GloVe

Classifiers	**Accuracy**	**Precision**	**Recall**	**F1-Score**
XGBoost	0.96	0.94	0.98	0.96
CatBoost	0.97	0.96	0.98	0.97
Stacking	0.97	0.96	0.98	0.97
Random Forest	0.98	0.98	0.99	0.99

Table IV compares the performance of different classifiers using the GloVe feature extraction technique. Across all metrics, Random Forest achieves the highest scores with 98% accuracy, 98% precision, 99% recall, and 99% F1-Score. This underscores Random Forest's effectiveness in classifying instances based on GloVe embeddings. Finally, Random Forest is identified as the best classifier for text data.

Comparision between Classifiers:

The study examines the performance of four classifiers: Random Forest, XGBoost, CatBoost, and Stacking. Additionally, it explores feature extraction techniques including TF-IDF, Count Vectorization, Word2Vec, and GloVe.Among these, Random Forest achieves the highest accuracy of 99% with TF-IDF feature extraction.

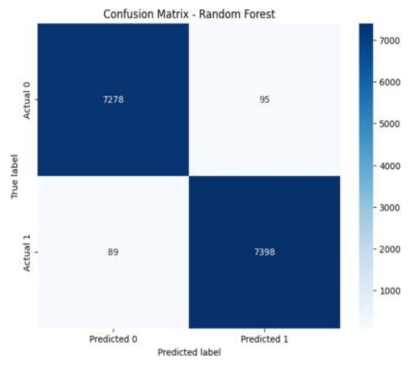

Fig. 5. confusion matrix of twitter data produced by Random Forest

The Random Forest classifier categorizes the Twitter dataset into two classes, and the predicted values are displayed in the form of a confusion matrix Fig 5.

The study thoroughly examined numerous feature extraction methods and text data categorization systems. When it came to extracting the most insightful and pertinent features from the text data, TF-IDF (Term Frequency-Inverse Document Frequency) performed the best out of all the feature extraction techniques that were tested. TF-IDF emphasizes terms that are unique to particular documents or classes by taking into account both the frequency of terms inside a document and their significance throughout the whole corpus.

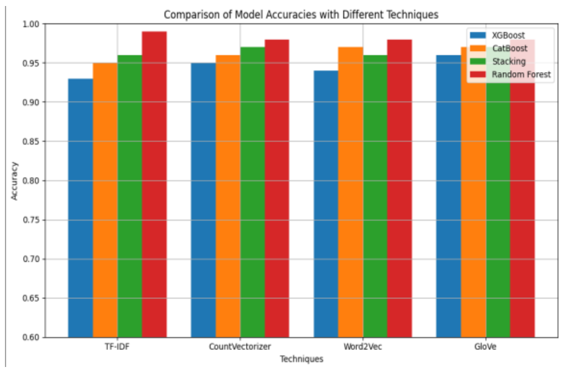

Fig. 6. Accuracy scores of classifiers using different feature extractions

To find the most accurate model for text classification tasks, the study assessed many classification methods concurrently. One flexible ensemble learning method that performed quite well with text data was Random Forest with the Accuracy of 99%.

V. Conclusion

The study compared the performance of Random Forest, XGBoost, CatBoost, and Stacking algorithms for sentiment analysis. Through rigorous evaluation, it was observed that Random Forest achieved the highest accuracy of 99% among the four algorithms tested, indicating its effectiveness in accurately predicting sentiment from the data. Furthermore, this study employed NLTK sentiment analysis as a benchmark comparison. Interestingly, the results revealed that NLTK sentiment analysis yielded even higher accuracy compared to the machine learning algorithms considered, highlighting the robustness and efficiency of NLTK in capturing sentiment from textual data.Overall, TF-IDF emerged as the best feature extraction for text data , while Random Forest emerged as one of the top-performing machine learning algorithms, the superior accuracy achieved by NLTK sentiment analysis underscored its efficiency as a powerful tool for sentiment analysis tasks.

References

[1] D.C. Edara, L.P. Vanukuri, and V. et al. Sistla. "Sentiment analysis and text categorization of cancer medical records with LSTM". In: *J Ambient Intell Human Comput* 14 (2023), pp. 5309–5325. DOI: 10.1007/s12652-019-01399-8.

[2] Shihab Elbagir and Jing Yang. "Twitter sentiment analysis using natural language toolkit and VADER sentiment". In: *Proceedings of the international multiconference of engineers and computer scientists*. Vol. 122. 16. sn. 2019.

[3] Geetika Gautam and Divakar Yadav. "Sentiment analysis of twitter data using machine learning approaches and semantic analysis". In: *2014 Seventh international conference on contemporary computing (IC3)*. IEEE. 2014, pp. 437–442.

[4] Bhumika Gupta et al. "Study of Twitter sentiment analysis using machine learning algorithms on Python". In: *International Journal of Computer Applications* 165.9 (2017), pp. 29–34.

[5] Md Rakibul Hasan, Maisha Maliha, and M Arifuzzaman. "Sentiment analysis with NLP on Twitter data". In: *2019 international conference on computer, communication, chemical, materials and electronic engineering (IC4ME2)*. IEEE. 2019, pp. 1–4.

[6] Olga Kolchyna et al. "Twitter sentiment analysis: Lexicon method, machine learning method and their combination". In: *arXiv preprint arXiv:1507.00955* (2015).

[7] Akrivi Krouska, Christos Troussas, and Maria Virvou. "The effect of preprocessing techniques on Twitter sentiment analysis". In: *2016 7th international conference on information, intelligence, systems & applications (IISA)*. IEEE. 2016, pp. 1–5.

[8] Bac Le and Huy Nguyen. "Twitter sentiment anal- ysis using machine learning techniques". In: (2015), pp. 279–289.

[9] Lokesh Mandloi and Ruchi Patel. "Twitter sentiments analysis using machine learninig methods". In: *2020 International Conference for Emerging Technology (INCET)*. IEEE. 2020, pp. 1–5.

[10] MS Neethu and R Rajasree. "Sentiment analysis in twitter using machine learning techniques". In: *2013 fourth international conference on computing, communications and networking technologies (ICCCNT)*. IEEE. 2013, pp. 1–5.

[11] Adyan Marendra Ramadhani and Hong Soon Goo. "Twitter sentiment analysis using deep learning methods". In: *2017 7th International annual engineering seminar (InAES)*. IEEE. 2017, pp. 1–4.

[12] Varsha Sahayak, Vijaya Shete, and Apashabi Pathan. "Sentiment analysis on twitter data". In: *International Journal of Innovative Research in Advanced Engineering (IJIRAE)* 2.1 (2015), pp. 178–183.

[13] Nikhil Yadav et al. "Twitter sentiment analysis using supervised machine learning". In: *Intelligent data communication technologies and internet of things: Proceedings of ICICI 2020*. Springer. 2021, pp. 631–642.

[14] Sheresh Zahoor and Rajesh Rohilla. "Twitter Sentiment Analysis Using Machine Learning Algorithms: A Case Study". In: *2020 International Conference on Advances in Computing, Communication Materials (ICACCM)*. 2020, pp. 194–199. DOI: 10.1109/ICACCM50413.2020.9213011.