JURNAL INFORMATIKA
PROXIES

Vol.9, No.1, July, 2025

Comparison Between Random Forest And Xgboost Performance In Text Classification For Emotion Detection

Jessica Angela Lukito[1], Shinta Estri Wahyuningrum[2]

[1] Teknik Informatika Fakultas Ilmu Komputer, Universitas Katolik Soegijapranata, jsi03angela@gmail.com
[2] Teknik Informatika Fakultas Ilmu Komputer, Universitas Katolik Soegijapranata, shinta@unika.ac.id

Corresponding Author Email: shinta@unika.ac.id

https://doi.org/10.24167/proxies.v9i1.13090

Received: 2025-01-24
Revised: 2025-07-25
Accepted: 2025-10-22
Available online: 2025-10-23

Keywords:
Emotion Detection, Decision Tree, Random Forest, XGBoost

ABSTRACT

Humans can not read minds. In this era, where most people are using text-based communication through social media which are non-Face-to-Face interactions. A lot of misunderstandings happened during online conversations like texting because of unclear messages that leads to confusion. Unfortunately, the misunderstanding of a message can cause many negative things to happen such as fight, separation and many more. To resolve this issue, many research has been done by researchers. In some research, several researchers said that Random Forest is the best algorithm for text classification, while others said that XGBoost which is part of Decision Tree is the best. Moreover, there is a study that said Decision Tree is the worst performing algorithm for text classification. With this study, Random Forest and XGBoost as part of Decision Tree will be compared with several pre-processing scenarios and methods. Dataset used for this study is obtained from the Kaggle website which contains 416,809 unique values of sentences.

1. INTRODUCTION

Background

In this modern era, social media have serious impact on everyone’s life. However, social media usages have some limitations and disadvantages. Most of its interaction are through text, such as commenting, quoting, chatting, etc. Therefore, a lot of misunderstandings could happen within text interactions because text does not have definite tone, expression, feeling, etc.

To reduce this issue, many researchers implemented several algorithms in text classification topic. However, many confuse occurs as many external factors may affect the algorithm performance, such as unclean data, incomplete data, empty data etc. In this study we will implement two pre-processing methods (Case Folding and Stop-word Removal) and two extraction methods (TF-IDF and Count Vectorization).

Scope

In this study, Random Forest and XGBoost algorithms will be used to solve emotion detection through text issues. Random Forest and XGBoost will be compared from accuracy, precision, recall and F-score. In this research, two pre-processing and two extraction methods will be used and data set used is obtained from Kaggle website. The result of this study will lead to which algorithm have the best performance based on accuracy, precision, recall, and F-score for emotion detection in text.

Objective

The objective of this research is to compare Random Forest algorithm and XGBoost algorithm to observe which algorithm have higher percentages of accuracy, precision, recall and F-score in text classification for emotion detection based on the following preprocessing and extractions methods.

LITERATURE STUDY

Susandri S et al [1] researched using several algorithms for text classification and sentiment labeling which made the conclusion the result of both Random Forest and Decision Tree algorithms perform significantly better than other models, reaching an accuracy of 89%. Dataset used for this study was acquired from Alumni94 WhatsApp Group conversation. The WhatsApp Group was created on 19 January 2019, and the data used ranged from 20 January 2021 until 26 March 2022 with member of 132 persons and 2563KB of file size. This study revealed that Random Forest achieved higher precision and recall (83% and 64%, respectively). While Decision Tree had 80% and 66%. But the precision of the Decision Tree algorithm was 0.03 points higher than Random Forest.

Other researchers, El Bayeh et al [2] using Support Vector Machine, Naïve Bayes, KNN, Decision Tree, Random Forest and Logistic Regression came with conclusion of Support Vector Machine showed better performance in many scenarios, while Decision Tree showed the worst performance in most scenarios. However, Support Vector Machines have some limitations, several of them occupy more memory and training time consumes a lot of time.

In other study, Occhipinti A et al [3] have done some research with several algorithms, namely Naïve Bayes, SVM, KNN, Logistic Regression, Random Forest and XGBoost which come to result of XGBoost as the best performing classifier algorithm. Achieving 94% of F-score and Precision, 93% of Recall with 0.017 s of time consumption. While Random Forest reached an equal result of 94% of F-score, Precision and Recall with 0.025 s of time consumption. The dataset used for this study was the Enron Spam Corpus that contained 17,171 spam emails.

While Arwa Alshamsi et al [4] conducted a research sentiment analysis with Naïve Bayes, Decision Tree, Forest, KNN, Random Tree, and ID3 on social media. The data used for the research is 14,000 tweets about six American Airlines Companies. The research has made the conclusion that Naïve Bayes, Decision Tree, and ID3 gave better results on a balanced dataset. While on the other hand, KNN, Random Forest, Decision Tree, and Random Tree gave better results on unbalanced dataset.

Soni S et al [5] also conduct research in text classification. Using CNN based, namely TextConvoNet, Soni S et al compared the seven different algorithms to validate the research. Multinomial Naïve Bayes, Decision Tree, Random Forest, SVC, Gradient Boosting, KNN, and XGBoost are the seven algorithms used. There are five different datasets used for this research, namely Binary SST-2, Amazon Review for Sentiment Analysis Dataset, R8 Dataset, Twitter User Airline Sentiment, and Covid Tweets. In this study, researchers used eight performance measures namely, accuracy, precision, recall, F-score, specificity, G-means, and MCC. The conclusion of this research is that TextConvoNet outperformed all other algorithms on multiclass datasets and equal to with other algorithms on binary datasets.

In another study, with the same algorithm, CNN was written by Kamran Kowsari et al [6]. In the study, he used several preprocessing methods, namely text cleaning and pre-processing, syntactic word representation, for pre-processing methods and weighted words, word embedding for text feature extraction methods. The research has made the conclusion that the understanding of feature extraction methods can make text classification algorithms perform more efficiently. Although, pre-processing methods such as text cleaning could improve algorithm accuracy and robustness.

Using a different algorithm, Pranckevičius T et al [7] conducted research in text classification. Comparing Naïve Bayes, Random Forest, Decision Tree, SVM, and Logistic Regression algorithm with conclusion of Decision Tree has achieved the lower accuracy percentages (min: 24.10%, max: 34.58%) among other algorithms. However, Naïve Bayes has outperformed other algorithms with 1-2% higher percentages of accuracy than Random Forest and SVM algorithms.

In another study using the Decision Tree algorithm, Palad E et al [8] compared several Decision algorithms. Dataset used for this study obtained from Police incidents reports which contains 49,822 words of online scam data which mainly wrote in Filipino. In this study, Random Forest achieved the highest accuracy with percentages of 95.51% and second best is J48 with 92.65%. This study shows that even Random Forest have the higher percentages of accuracy even though Random Forest used more time than other algorithms to build models.

With the same Methods, Decision Tree, compared with Naïve Bayes and KNN was written by Ababneh J [9]. In this study, the author used 1,562 Arabic articles which were collected from the Saudi Press Agency. This study found that Decision Tree algorithms have the worst performance among two other algorithms with 82.3% of precision, 81.7% recall, and 82% F-Score. While Naïve Bayes perform better than those two other algorithms with 88.1% of precision, 87.9% of recall, and 88% of F-Score. However, those three algorithms used in this study give relatively high percentages of results in text classification.

Therefore, other researchers, Prachi N et al [10] held a study of Logistic Regression, Naïve Bayes, SVM, and Decision Tree algorithms. This study has made the conclusion of Decision Tree as the best performance algorithm based on accuracy, precision, recall, and F-Score percentages with the following 90%, 89%, 89%, 90%. However, Deep Learning algorithms, namely LSTM still have higher percentages of accuracy, precision, recall, and F-Score with 94%, 95%, 94%, 95%.

Based on several literature above, Random Forest and Decision Tree algorithms have several different conclusions, which are baffling. With this study, research will be arranged to compare Random Forest and Decision Tree algorithm with several pre-processing methods, Decision Tree algorithm chosen are XGBoost, as Occhipinti A et al [3] has found that XGBoost is having best performance as a classifier.

2. RESEARCH METHOD

Dataset Collection

Dataset for this study was taken from Kaggle.com. The dataset contained 416,809 unique values of sentences which were taken from Twitter. The dataset is in CSV format and already labeled. The dataset contains three columns, namely number, text and label. Dataset will be splitted for training model and validation model. Dataset can be found and downloaded online via this link https://www.kaggle.com/datasets/nelgiriyewithana/emotions/data.[1] This dataset will be used for all processes in this study and the variables used are text and label.

Data Pre-Processing

This study will use several steps of data processing. The dataset is already labeled with six sentiments which are sad (0), joy (1), love (2), anger (3), fear (4), surprise (5). The data processing will be using two scenarios:

1. Scenario 1: Case Folding. This process is to replace uppercase to lowercase and remove any non-alphabetic character.
2. Scenario 2: Stop word removal. This process is to remove commonly used words, e.g. is, a, the, or, etc.

Since the two scenarios have been executed, Word Feature Extraction will be done with two methods. The methods for extraction are Count Vectorization and TF-IDF. The next step is to try the processed dataset with XGBoost and Random Forest algorithm and the result of accuracy, precision, recall and F-score on each scenario will be evaluated.

Count Vectorization is a method to convert text in the dataset into a numeric format. Count Vectorization will also count the frequency of how often those words appear in the dataset.

And TF-IDF or also known as Term Frequency-Inverse Document Frequency is a method to measure how relevant are the words in the dataset. TF-IDF can be counted by using the formula below. The fifty samples of dataset are taken for the implementation calculation below:

$$TF = \frac{Frequency\ of\ word\ in\ the\ data}{Total\ number\ of\ words\ in\ the\ data}$$

$$IDF = log\frac{Total\ number\ of\ data}{Number\ of\ dataset\ containing\ the\ word}$$

$$TF - IDF = TF\ x\ IDF$$

While TF-IDF has a formula to implement, Count Vectorizer does not use any formula for its implementation. Count Vectorization replaces the string format dataset with numeric format. And will also count the frequency of how often those words appear in the dataset.

Random Forest Algorithm

Unlike Decision Tree, Random Forest algorithm is an algorithm that combines the result of multiple trees for its final result. So, basically Random Forest is a group of Decision Trees that have small differences of specialties. With these methods, Random Forest will help to solve more complex problems than one Decision Tree. The sample result of each tree will be taken as part of the result and will be combined with another tree's sample result to make a final result. The more trees used will increase the accuracy of the Random Forest model.

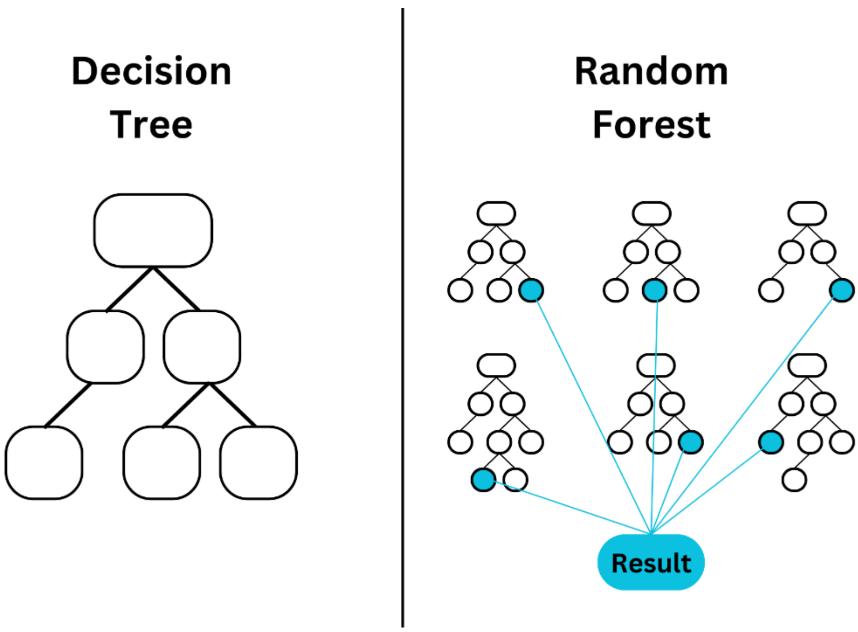

Figure 1. Decision Tree and Random Forest Algorithm Process

Random Forest will build a Decision Tree for each subset then train the data randomly for each Decision Tree. For building a Decision Tree, the first split will be done by using Gini Impurity. Using Gini Impurity, the split for Random Forest will be calculated.

XGBoost Algorithm

The comparison of Random Forest and XGBoost will be done after the dataset is processed with pre-processing methods above. Each algorithm will also apply two different scenarios. The results of each algorithm and each scenario will be used for the validation model of the algorithms.

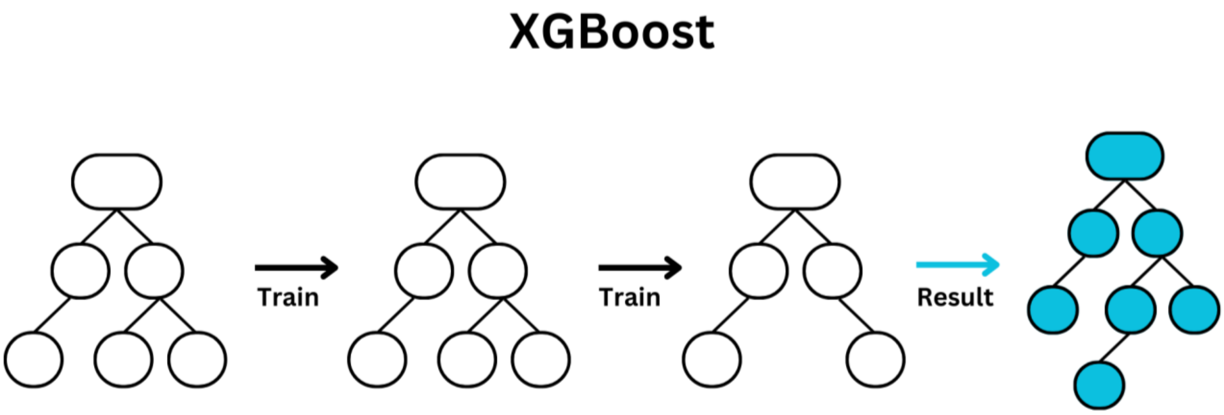

Figure 2. XGBoost Algorithm Process

While Random Forest is collecting samples of each tree's results as its final result, XGBoost takes all results and trains each tree to maximize the final result of the algorithm. XGBoost will calculate each error of the previous model then will add predictions based on errors before to increase its accuracy. For each error or mis-classification that happens in the previous tree will boost the next tree's predictions.

IMPLEMENTATION AND RESULT

Implementation

The implementation of Random Forest and XGBoost comparison of accuracy, precision,

recall and f-score will be explained in this section. In this research, significant tests will be done with 4 rounds. First and second round will be done with 80% of data for training and 20% for validation. Third and fourth round will use 50% of data for training and 50% for validation. This significant test is done so that the final result can be drawn from the average round results.

Result

In this research, two pre-processing methods (Case Folding and Stopword removal) and two extraction methods (TF-IDF and Count Vectorizer) have been implemented to the Random Forest and XGBoost algorithm. The result of these comparisons will be compared in this section, below will be shown the diagram for Random Forest and XGBoost validation results. The diagram below showed that XGBoost algorithm outperform Random Forest algorithm in all Random Forest combinations.

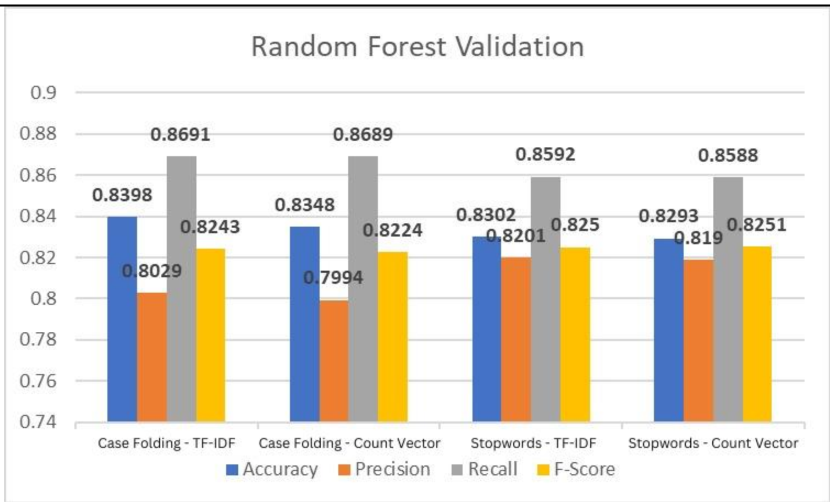

Figure 3. Random Forest Validation Results Diagram

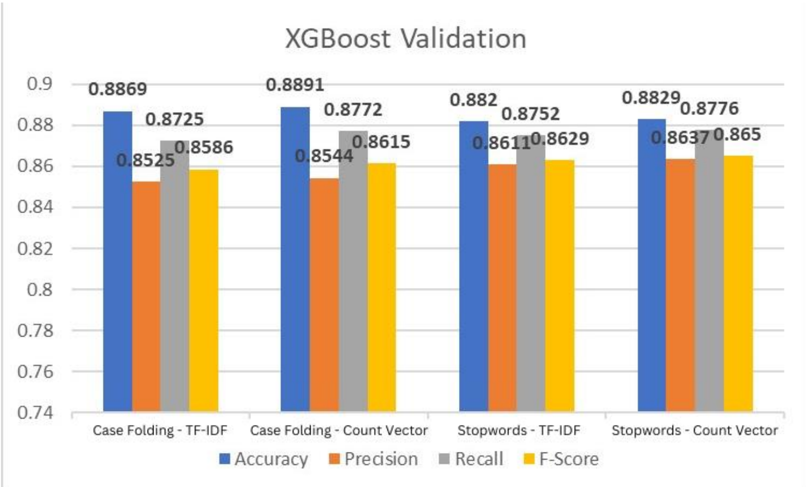

Figure 4. XGBoost Validation Result Diagram

Discussion

In this study, the results from eight algorithm comparisons experiments show the impact of pre-processing techniques and extraction methods on the performance of Random Forest and XGBoost models. The experiments used two pre-processing scenarios, which are case folding and stopwords removal and two extraction methods which are TF-IDF and Count Vectorizer to compare both algorithms on classification tasks. The significant tests are also done in this experiment with 4 rounds. First and second round will be done with 80% of data for training and 20% for validation. Third and fourth round will use 50% of data for training and 50% for validation. Note that the results are obtained from an average of 4 rounds of experiments.

The Random Forest model with Case Folding pre-processing scenario combined with TF-IDF as extraction method scored an accuracy of 0.8436 on training and 0.8398 on the validation, precision score of 0.8069 for training and 0.8029 for validation, recall score of 0.8742 for training and 0.8691 for validation, and F1-Score of 0.8287 for training and 0.8243 for validation. This combinations of Random Forest with Case Folding as pre-processing method and TF-IDF as extraction method is showing the highest performance result compared to other combination for Random Forest algorithm. This may indicate that in this study Random Forest has high compatibility with Case Folding and TF-IDF. This result is also supported by a study written by Imane Lasri et al [11] using Random Forest and TF-IDF as extraction methods and comparing several algorithms including Random Forest. The result of his study showed that Random Forest coupled with TF-IDF gives the highest performance result among other algorithms with an accuracy score of 98%.

The XGBoost model with Case Folding as pre-processing method and Count Vectorizer as extraction method gives a very slightly higher result compared to other XGBoost and Random Forest cases. With accuracy score of 0.8979 for training and 0.8891 for validation, precision score of 0.8650 for training and 0.8544 for validation, recall score of 0.8900 for training and 0.8772 for validation, and F1-score of 0.8729 for training and 0.8615 for validation. This result of XGBoost indicates that XGBoost outperformed Random Forest algorithm in classification cases in this study and also combination of XGBoost algorithm with Case Folding as pre-processing method and Count Vectorizer as extraction method outperform all other combinations. This result is also supported in the research that stated that XGBoost combined with Count Vectorizer as extraction method gives the highest result among all other cases by giving highest accuracy of 80.89% in the study written by Rishickesh R et al [12].

The results indicate that XGBoost consistently outperforms Random Forest in both pre-processing scenarios. The study conducted by Occhipinti A et al [3] also agreed with XGBoost as the best performing algorithm compared to Random Forest algorithm. The best performance is XGBoost with Case Folding as pre-processing method and Count Vectorizer as extraction method, which resulted in the highest performance. However, for Random Forest, combination of Case Folding as pre-processing method and TF-IDF as extraction method gives the highest performance compared to other Random Forest combinations. This indicates that the choice of pre-processing methods and extraction methods can significantly impact the model performances.

CONCLUSION

The results from the eight cases indicate the importance of the choice for algorithms, pre-processing methods and extraction methods combinations in order to achieve optimal performance for classification cases. XGBoost consistently performed better than Random Forest in all scenarios, with the combination of Case Folding and Count Vectorizer, XGBoost making the highest result of accuracy score of 0.8979 for training and 0.8891 for validation, precision score of 0.8650 for training and 0.8544 for validation, recall score of 0.8900 for training and 0.8772 for validation, and F1-score of 0.8729 for training and 0.8615 for validation. For XGBoost models, the result for all combination cases only has a slight difference, this may also indicate that the XGBoost model has already reached its maximum performance.

For the Random Forest algorithm, the best combination is Random Forest with Case Folding as pre-processing method and TF-IDF as extraction method. With accuracy score of 0.8436 on training and 0.8398 on the validation, precision score of 0.8069 for training and 0.8029 for validation, recall score of 0.8742 for training and 0.8691 for validation, and F1-Score of 0.8287 for training and 0.8243 for validation. This result of the Random Forest algorithm also indicates that XGBoost outperformed Random Forest algorithm in all combination cases. This may also indicate that in this study, XGBoost is the best algorithm for text classification tasks compared to Random Forest algorithm.

However, these results are also dependent on the hyperparameter configurations values used for the experiments. The hyperparameters used for Random Forest and XGBoost were set for this case, making the comparison valid for the specific parameter settings used. If different hyperparameter values were used, the results could also be different, the hyperparameters used in this study are examples of hyperparameters that are effective for each scenario.

In summary, while the experiments provide valuable insights into the comparison of accuracy, precision, F-score and recall of Random Forest and XGBoost with Case Folding and Stopwords removal pre-processing methods and TF-IDF and Count Vectorizer, the conclusions must include the specific hyperparameter settings used. For suggestions of future work, researchers could explore the use and impact of hyperparameter optimization for further research

REFERENCES

[1] S. Susandri, S. Defit, and M. Tajuddin, “SENTIMENT LABELING AND TEXT CLASSIFICATION MACHINE LEARNING FO REFERENCES

[1] S REFERENCES

[1] S. Susandri, S. Defit, and M. Tajuddin, “SENTIMENT LABELING AND TEXT CLASSIFICATION MACHINE LEARNING FOR WHATSAPP GROUP,” JITK (Jurnal Ilmu Pengetahuan dan Teknologi Komputer), vol. 9, no. 1, 2023, doi: 10.33480/jitk.v9i1.4201

[2] A. Palanivinayagam, C. Z. El-Bayeh, and R. Damaševičius, “Twenty Years of Machine-Learning-Based Text Classification: A Systematic Review,” Algorithms, vol. 16, no. 5, 2023, doi: 10.3390/a16050236

[3] A. Occhipinti, L. Rogers, and C. Angione, “A pipeline and comparative study of 12 machine learning models for text classification,” 2022. doi: 10.1016/j.eswa.2022.117193

[4] A. Alshamsi, R. Bayari, and S. Salloum, “Sentiment analysis in English Texts,” Advances in Science, Technology and Engineering Systems, vol. 5, no. 6, 2020, doi: 10.25046/AJ0506200

[5] S. Soni, S. S. Chouhan, and S. S. Rathore, “TextConvoNet: a convolutional neural network based architecture for text classification,” Applied Intelligence, vol. 53, no. 11, 2023, doi: 10.1007/s10489-022-04221-9

[6] K. Kowsari, K. J. Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, “Text classification algorithms: A survey,” Information (Switzerland), 2019. doi: 10.3390/info10040150

[7] T. Pranckevičius and V. Marcinkevičius, “Comparison of Naive Bayes, Random Forest, Decision Tree, Support Vector Machines, and Logistic Regression Classifiers for Text Reviews Classification,” Baltic Journal of Modern Computing, vol. 5, no. 2, 2017, doi: 10.22364/bjmc.2017.5.2.05

[8] E. B. B. Palad, M. J. F. Burden, C. R. Dela Torre, and R. B. C. Uy, “Performance evaluation of decision tree classification algorithms using fraud datasets,” Bulletin of Electrical Engineering and Informatics, vol. 9, no. 6, 2020, doi: 10.11591/eei.v9i6.2630

[9] J. Ababneh, “Application of Naïve Bayes, Decision Tree, and K-Nearest Neighbors for Automated Text Classification,” Mod Appl Sci, vol. 13, no. 11, 2019, doi: 10.5539/mas.v13n11p31

[10] N. N. Prachi, M. Habibullah, M. E. H. Rafi, E. Alam, and R. Khan, “Detection of Fake News Using Machine Learning and Natural Language Processing Algorithms,” Journal of Advances in Information Technology, vol. 13, no. 6, 2022, doi: 10.12720/jait.13.6.652-661

[11] I. Lasri, A. Riadsolh, and M. Elbelkacemi, “Real-time Twitter Sentiment Analysis for Moroccan Universities using Machine Learning and Big Data Technologies,” International Journal of Emerging Technologies in Learning, vol. 18, no. 5, 2023, doi: 10.3991/ijet.v18i05.35959

[12] R. Rishickesh, R. P. Ram Kumar, A. Shahina, and A. Nayeemullah Khan, “Identification of duplication in questions posed on knowledge sharing platform quora using machine learning techniques,” International Journal of Innovative Technology and Exploring Engineering, vol. 8, no. 12, 2019, doi: 10.35940/ijitee.L3017.1081219. Susandri, S. Defit, and M. Tajuddin, “SENTIMENT LABELING AND TEXT CLASSIFICATION MACHINE LEARNING FOR WHATSAPP GROUP,” JITK (Jurnal Ilmu Pengetahuan dan Teknologi Komputer), vol. 9, no. 1, 2023, doi: 10.33480/jitk.v9i1.4201