Performance Analysis of SMOTE and ADASYN in Stunting Disease Classification with the XGBoost Algorithm

Christian Sri Kusuma Aditya a)

Department of Informatics, Universitas Muhammadiyah Malang, Malang, Indonesia

*a) Corresponding author: christianskaditya@umm.ac.id*

**Abstract.** Stunting is a major public health issue characterized by impaired growth and development in children, often resulting from chronic malnutrition. Accurate classification of stunting cases is essential for early intervention and effective health policy implementation. However, imbalanced datasets, where the number of stunting cases is significantly lower than non-stunting cases, pose a challenge to machine learning models. This study evaluates the performance of two oversampling techniques, Synthetic Minority Oversampling Technique (SMOTE) and Adaptive Synthetic Sampling (ADASYN), in improving classification results for stunting disease using the XGBoost algorithm. A stunting dataset was preprocessed and subjected to both SMOTE and ADASYN to address class imbalance. The XGBoost classifier was then trained and evaluated using accuracy, precision, recall, and F1-score as performance metrics. The results indicate that both SMOTE and ADASYN improve classification performance compared to the original imbalanced dataset, with ADASYN showing significantly higher precision, suggesting better handling of minority class detection. These findings highlight the importance of appropriate data balancing techniques in enhancing machine learning models for health-related classification tasks, particularly in stunting detection.

**Keywords:** SMOTE, ADASYN, Stunting, XGBoost, Classification

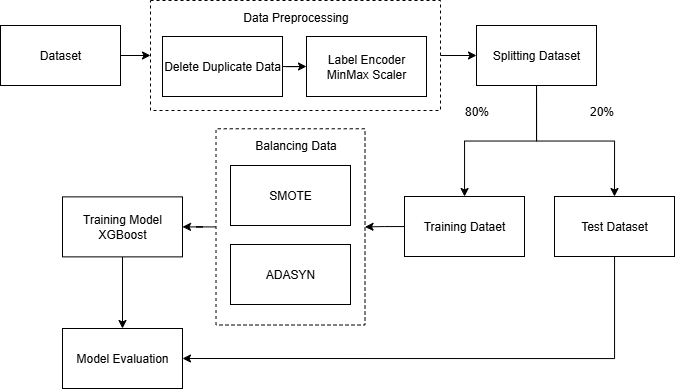
# INTRODUCTION

Stunting is a failure of growth and development in toddlers (babies under five years old) due to chronic malnutrition, so that their height becomes shorter than it should be according to their age [1][2][3]. One of the impacts of stunting is that children have a low level of intelligence and are susceptible to disease, especially in children who experience stunting when they are under five and two years old [4]. Many factors influence stunting including culture, education, access to health care, economic status and political climate, food quality and agricultural systems, sanitation, and environmental conditions all play a role, as an influence of the internal environment, in the child's household must be considered such as good childcare, exclusive breastfeeding (ASI), providing the best complementary foods for breast milk (MPASI), and immunization [5]. Based on the results of the Indonesian Nutritional Status Survey (SSGI) in 2022, the stunting rate in Indonesia is still relatively high at 21.6%. Although this figure has decreased from 24.4% in 2021, significant efforts are still needed to achieve the target of reducing stunting to 14% by 2024 [6]. One effort that can be made to achieve the target of reducing stunting is through early detection of stunting [7].

Machine learning offers an innovative approach to early stunting detection, enabling the use of various toddler data attributes to predict stunting levels. This is necessary to identify toddlers at risk of stunting so that appropriate preventive measures and interventions can be implemented. Machine learning can improve the accuracy of stunting detection, helping health practitioners and policymakers target interventions more effectively, ultimately reducing the prevalence of stunting and avoiding its long-term negative impacts. Therefore, the application of machine learning to address stunting is highly relevant and urgent [7]. As a developing field of artificial intelligence, machine learning offers innovative approaches to stunting detection [8]. One such approach is the use of classification. The advantage of machine learning in detecting stunting lies in its potential to improve accuracy, efficiency, and speed of detection. The application of machine learning algorithms can help health practitioners and policymakers target interventions more effectively, ultimately reducing the prevalence and impact of stunting [9]. Several studies have been conducted by other researchers in 2024 to support this research. Indah Ardhia Cahyani and others [10] proposed research on the use of Multilayer Perceptron with GridSearchCV hyperparameter tuning to classify stunting. This study used a dataset from the Kaggle website derived from Harnelia's 2024 study entitled "Stunting Factors." This dataset contains 10,000 data points on the anthropometrics of children under five years old. This study produced an accuracy value of 82.37%. Furthermore, another study conducted by Muhammad Fikri [11] in the same year used the XGBoost algorithm to classify stunting with the same dataset. This study successfully achieved an accuracy of 86%. Both studies share the same weakness: relying solely on the performance of the machine learning algorithm used without prior data balancing, while the dataset used is imbalanced, where this class imbalance can limit model performance [12]. Based on previous studies, it can be concluded that although machine learning methods such as Multilayer Perceptron and XGBoost can produce quite good accuracy, unaddressed class imbalance can limit model performance. Therefore, this study aims to analyze the performance of various data balancing methods such as SMOTE and ADASYN in classifying stunting diseases using the XGBoost algorithm where XGBoost excels if the dataset is processed in numerical form or has been encoded manually.

# METHODS

This study uses data balancing techniques such as SMOTE and ADASYN with the XGBoost algorithm to classify stunting. To classify data from the dataset, several classification stages are required, as described in the flowchart in Figure 1.



**Figure 1.** Research Method

## Dataset

The dataset used in this study is the same as the dataset used in previous studies [10], [11]. The dataset is taken from Kaggle with the dataset title "Stunting Factors". The dataset contains 8 attributes and 10,000 entities. The target column of this dataset refers to the presence of stunting. A description of each attribute is in Table 1, and the visualization of the first five data in the dataset used in this study is in Table 2.

**Table 1**. Dataset Description

|  |  |
| --- | --- |
| Attribute | Description |
| Gender | Gender consists of 2 values, male or female |
| Age | Age in months |
| Birth Weight | Birth weight in kg |
| Birth Length | Birth length in cm |
| Body Weight | Current body weight in cm |
| Body Length | Current length in cm |
| Breastfeeding | “Breastfeeding or “Not” |
| Stunting | “Stunting” or “Not” |

**Table 2**. Sampling Dataset

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Gender | Age | Birth Weught | Birth Length | Body Weight | Body Length | Breastfeeding | Stunting |
| Male | 17 | 3.0 | 49 | 10.0 | 72.2 | No | No |
| Female | 11 | 2.9 | 49 | 2.9 | 65.0 | No | Yes |
| Male | 16 | 2.9 | 49 | 8.5 | 72.2 | No | Yes |
| Male | 31 | 2.8 | 49 | 6.4 | 63.0 | No | Yes |
| Male | 15 | 3.1 | 49 | 10.5 | 49.0 | No | Yes |

## Data Preprocessing

Data preprocessing is a crucial step to ensure the data used in the analysis model is optimally maintained and free from errors that could impact the research results. At this stage, the first step is to remove duplicate data. Duplication can occur due to recording errors or the merging of multiple data sources. By removing duplicate data, the dataset is cleaner and ensures that the model does not learn from repetitive information, which can lead to overfitting and affect the quality of the analysis. Next, the categorical data in the dataset needs to be converted into a numeric format to enable the algorithm to work effectively. This process is performed using Label Encoding, where each category is converted into a unique numeric value, the result of Label Encoder step can be seen in Table 3. This step is crucial because algorithms like CatBoost require numeric representation to process and generate predictions.

**Table 3**. Example of the Results of Applying the Label Encoder

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Gender | Age | Birth Weught | Birth Length | Body Weight | Body Length | Breastfeeding | Stunting |
| 1 | 17 | 3.0 | 49 | 10.0 | 72.2 | 0 | 0 |
| 0 | 11 | 2.9 | 49 | 2.9 | 65.0 | 0 | 1 |
| 1 | 16 | 2.9 | 49 | 8.5 | 72.2 | 0 | 1 |
| 1 | 31 | 2.8 | 49 | 6.4 | 63.0 | 0 | 1 |
| 1 | 15 | 3.1 | 49 | 10.5 | 49.0 | 0 | 1 |

The use of Min-Max Scaler becomes particularly relevant when a dataset contains features with widely varying value ranges. For example, one dataset might have a feature measured in centimeters, while another feature is measured in kilograms. This difference in scale can make it difficult for algorithms, especially gradient-based ones like CatBoost, to achieve optimal stability. By applying Min-Max Scaler, the training process becomes more stable and efficient, as each feature contributes equally to the final model result. This technique is simple yet highly effective in improving the performance of machine learning algorithms. The following is an example of the results of applying Min-Max Scaler to this research dataset, which can be seen in Table 4.

**Table 4**. Example of the Results of Applying Min-Max Scaler

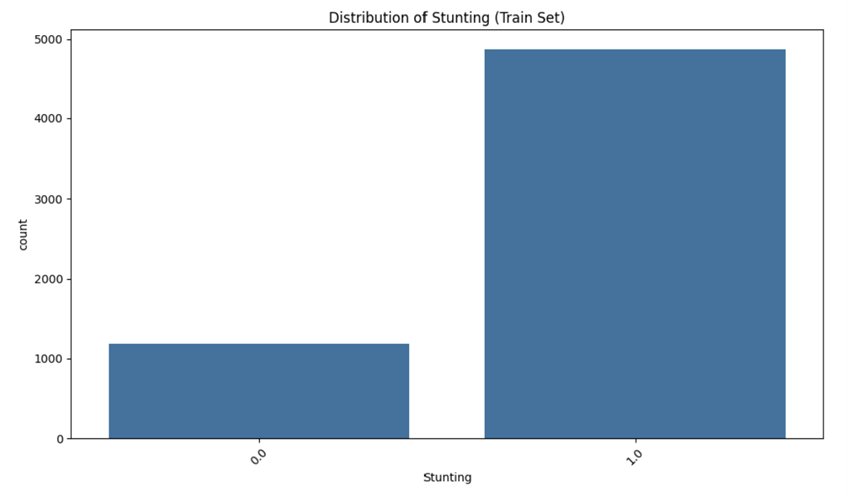
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Gender | Age | Birth Weught | Birth Length | Body Weight | Body Length | Breastfeeding | Stunting |
| 1.0 | 0.26 | 0.91 | 0.5 | 0.93 | 0.53 | 0.0 | 0.0 |
| 0.0 | 0.11 | 0.81 | 0.5 | 0.00 | 0.36 | 0.0 | 1.0 |
| 1.0 | 0.23 | 0.81 | 0.5 | 0.73 | 0.53 | 0.0 | 1.0 |
| 1.0 | 0.59 | 0.72 | 0.5 | 0.46 | 0.32 | 0.0 | 1.0 |
| 1.0 | 0.21 | 1.00 | 0.5 | 1.00 | 0.00 | 0.0 | 1.0 |

## Splitting Dataset

In this study, the dataset was split into two: 80% train data and 20% test data. The goal was to separate the data used to train the model from the data used to evaluate the model. This separation ensures that the model evaluation reflects its performance on previously unseen data, thereby reducing the risk of overfitting. This dataset splitting was performed before applying the data balancing technique to ensure that the balancing was only applied to the train data. This approach was designed to prevent information leakage from the test data, which could result in invalid model evaluations.

## Balancing Dataset

To address the imbalanced class distribution in this research dataset, various data balancing techniques were applied, such as SMOTE and ADASYN. These techniques work by increasing the number of samples in the minority class either by creating synthetic data or duplicating existing samples. By balancing the data, the model can learn more fairly and produce more accurate predictions for all classes in the dataset. In this study, data balancing was only performed on the train data, so that the test data is the original data from the dataset (not synthetic or artificial data from oversampling). The visualization of the target class distribution in this research train data can be seen in Figure 2, which shows that class 0.0 (NonStunting) numbered 1185 and class 1.0 (Stunting) numbered 4873.



**Figure 2.** Comparison of Class Distributions in Datasets

In this study, both SMOTE and ADASYN were applied to the training data to balance the target class distribution. The target class distribution in the training data, which initially consisted of 6,058 data sets, 1,185 for class 0 (Non-Stunting) and 4,873 for class 1 (Stunting), increased to 9,746 data sets that consisted 4,873 for class 0 (Non-Stunting) and 4,873 for class 1 (Stunting).

In the next stage, the researchers used the XGBoost algorithm to create a stunting disease classification model. To optimize model performance, the researchers adjusted several parameters using the CatboostClassifier class in the catboost library, as shown in Table 5.

**Table 5**. Parameter XGBoost

|  |  |
| --- | --- |
| Parameter | Value |
| iterations | 1000 |
| depth | 4 |
| learning\_ratee | 0.3 |
| loss\_function | logloss |
| verbose | 100 |
| early\_stopping\_rounds | 50 |

## Evaluation

The final stage of this research is an evaluation aimed at determining the accuracy of the developed model [21] on Javanese text data that has undergone stemming. This evaluation was conducted indirectly, by observing the impact of the stemming results on classification performance using the XGBoost method. Thus, the success of the method is measured by its contribution to improving classification accuracy, rather than by linguistic validity.

The evaluation metrics used include Precision, Recall, F1-Score, and Accuracy. Precision, as described in equation (4), is the proportion of positive data that is correctly predicted out of all data classified as positive. Recall, as formulated in equation (5), is the proportion of positive data successfully identified from all actual positive data. F1-Score, in (6), is the average obtained from the precision and Recall values. Meanwhile, Accuracy, as in equation (7), indicates the proportion of data correctly predicted from the total data tested.

The Confusion Matrix is used to improve model performance evaluation. This matrix provides a visual representation of the classification process prediction performance [30]. This method provides a detailed description of the model's classification performance by identifying four key values: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). TP is the number of positive data predicted correctly. TN is the number of negative data predicted incorrectly. FP is the number of positive data with incorrect values. FN is the number of negative data with correct values.

*Precision =* (5)

*Recall =* (6)

*F1-Score =* (7)

*Accuracy =* (8)

# RESULTS AND DISCUSSION

In the first scenario, data balancing was performed using the Synthetic Minority Oversampling Technique (SMOTE) before the model training process. SMOTE works by creating synthetic samples of the minority class based on the similarity between nearest neighbors. This technique aims to address class imbalance by adding artificial data, rather than simply copying existing data. After the data was balanced with SMOTE, the XGBoost model was trained with parameter settings of a learning rate of 0.3 and a tree depth of 4. The training process was stopped early by the overfitting detector at the 86th iteration because the model's performance on the test data no longer showed significant improvement after 50 consecutive iterations. The best loss value achieved on the test data was 0.34. After training, the model was shrunk to only the first 87 iterations. The evaluation results of the confusion matrix showed that 90 data were correctly predicted (True Positive) in the Non-stunting class, while 169 data were incorrectly predicted (False Positive) in the same class. For the Stunting class, 75 data points were predicted incorrectly (False Negative) and 1,181 data points were predicted correctly (True Negative). This evaluation can be seen in Figure 3.

# 

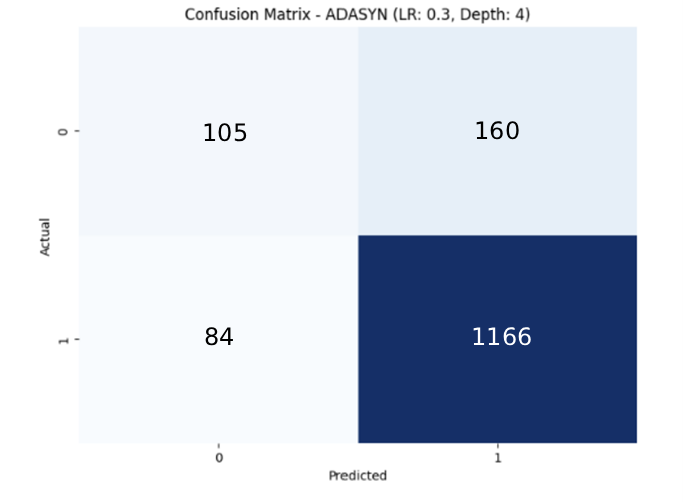
**Figure 3.** Confusion Matrix of SMOTE and XGBoost

After obtaining the confusion matrix value, the model evaluation results can be calculated to generate Precision, Recall, F1-Score, and Accuracy values of a model using the Classification Report available in the sklearn library as shown in Table 6. The Classification Report for the first scenario is in Table 6. Based on the evaluation results, this model produced an accuracy of 85%, indicating that the model is quite good at classifying stunting data after balancing using SMOTE.

**Table 6**. Performance of SMOTE and XGBoost Classification

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | F1-Score |
| Non-Stunting | 0.55 | 0.43 | 0.41 |
| Stunting | 0.89 | 0.91 | 0.87 |
| Macro AVG | 0.68 | 0.65 | 0.66 |
| Weighted AVG | 0.84 | 0.82 | 0.79 |

The second scenario uses the Adaptive Synthetic Sampling (ADASYN) technique. ADASYN is similar to SMOTE, but has an adaptive approach, giving more attention to minority data that are more difficult to classify. ADASYN dynamically determines the number of synthetic samples to be generated based on local distributions, thus helping the model learn better from complex patterns. After the balancing process with ADASYN, the XGBoost model was trained with the balanced data. The model was tested with the same test data as in the other scenarios. The training process was stopped early by the overfitting detector at the 78th iteration because the model's performance on the test data no longer showed significant improvement after 50 consecutive iterations. The best loss value achieved on the test data was 0.30. After training, the model was shrunk to only the first 79 iterations. The confusion matrix evaluation results showed that 105 data were correctly predicted (True Positive) in the Non-Stunting class, while 160 data were incorrectly predicted (False Positive) in the same class. For the Stunting class, there were 84 data points predicted incorrectly (False Negative) and 1,166 data points predicted correctly (True Negative). This evaluation can be seen in Figure 4.



**FIGURE 4.** Confusion Matrix of ADASYN and XGBoost

After obtaining the confusion matrix value, the model evaluation results can be calculated to produce Precision, Recall, F1-Score, and Accuracy values of a model using the Classification Report available in the sklearn library as shown in Table 7. Based on the evaluation results, this model produces an accuracy of 87%, which indicates that the model is good at classifying stunting data after balancing using ADASYN.

**Table 7**. Performance of ADASYN and XGBoost Classification

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | F1-Score |
| Non-Stunting | 0.65 | 0.42 | 0.47 |
| Stunting | 0.89 | 0.95 | 0.91 |
| Macro AVG | 0.76 | 0.69 | 0.69 |
| Weighted AVG | 0.84 | 0.82 | 0.84 |

Based on the performance evaluation results of the XGBoost model with two data balancing methods, namely SMOTE and ADASYN, it appears that the ADASYN method provides the best results overall. The model with ADASYN produces an accuracy of 87%, with a precision of 0.89, a recall of 0.95, and an F1-score of 0.91 for the stunting class. Although the performance for the non-stunting class is still relatively low (precision of 0.65, recall of 0.42), this result is still better than the other methods. Meanwhile, the SMOTE method provides almost similar performance, with an accuracy of 85% and an F1-score of 0.87 for the stunting class, but with a slight decrease in recall and precision for the non-stunting class. In general, these results indicate that the balancing method that produces synthetic data variations ADASYN is more effective when compared to SMOTE, especially in the context of classifying stunting diseases with unbalanced data where the precision of the Non-Stunting class produces a significant difference in values. The comparison of the results is shown in Table 8.

**Table 8**. Comparison of Results

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Balancing Data | Class | Precision | Recall | F1-Score | Accuracy |
| SMOTE | Non-Stunting | 0.55 | 0.43 | 0.41 | 0.85 |
|  | Stunting | 0.89 | 0.91 | 0.87 |
| ADASYN | Non-Stunting | 0.65 | 0.42 | 0.47 | 0.87 |
|  | Stunting | 0.89 | 0.95 | 0.91 |

# CONCLUSION

Stunting is influenced by various factors, including long-term nutritional deficiencies, suboptimal parenting, and social and environmental conditions that do not support child growth and development. Early detection of stunting is crucial to prevent long-term impacts, yet many people remain unaware of the importance of early screening. Based on the scenarios implemented in this study, it can be concluded that the implementation of the XGBoost algorithm in classifying stunting yields high accuracy. Furthermore, the use of the ADASYN data balancing method significantly improves model performance, particularly in one initially unbalanced class, compared to the SMOTE method. This study demonstrates that selecting the right balancing technique plays a crucial role in optimizing classification results on unbalanced data, particularly in the context of stunting prediction.

# References

1. V. C. M, R. Rudy, and D. S. Naga, "Fast and accurate spelling correction using trie and Damerau-levenshtein distance bigram," *TELKOMNIKA (Telecommunication Computing Electronics and Control),* vol. 16, no. 2, pp. 827-833, 2018, doi: 10.12928/TELKOMNIKA.v16i2.6890.

2. D. Soyusiawaty, A. H. S. Jones, and N. L. Lestariw, "The stemming application on affixed javanese words by using nazief and adriani algorithm," in *IOP Conference Series: Materials Science and Engineering*, 2020, vol. 771, no. 1: IOP Publishing, p. 012026, doi: 10.1088/1757-899X/771/1/012026.

3. R. A. Pratama, A. A. Suryani, and W. Maharani, "Part of Speech Tagging for Javanese Ngoko Language with Hidden Markov Model," J-Cosine(Journal of Computer Science and Informatics Engineering), vol. 4, doi: <https://doi.org/10.29303/jcosine.v4i1.346>.

4. D. E. Cahyani, L. M. T. Utami, and H. Setiadi, "Clustering of Javanese News in Krama Alus Level with Javanese Stemming," in 2019 International Conference on Information and Communications Technology (ICOIACT), 2019: IEEE, pp. 462-467.

5. N. Hidayatullah, A. P. Wibawa, and H. A. Rosyid, "Penerapan ECS Stemmer untuk Modifikasi Nazief & Adriani Berbahasa Jawa," *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi),* vol. 3, no. 3, pp. 343-348, 2019, doi: 10.29207/resti.v3i3.994.

6. S. I. Melia, J. Sholihah, D. Nisak, I. S. Juniaristha, and A. T. Ni'mah, "The Ngoko Javanese Stemmer uses the Enhanced Confix Stripping Stemmer Method," *Rekayasa,* vol. 16, no. 1, pp. 107-112, 2023.

7. A. P. Wibawa, F. A. Dwiyanto, I. A. E. Zaeni, R. K. Nurrohman, and A. Afandi, "Stemming javanese affix words using nazief and adriani modifications," *J. Inform,* vol. 14, no. 1, p. 36, 2020, doi: 10.26555/jifo.v14i1.a17106.

8. Z. Abidin and A. Junaidi, "Text Stemming and Lemmatization of Regional Languages in Indonesia: A Systematic Literature Review," *Journal of Information Systems Engineering and Business Intelligence,* vol. 10, no. 2, pp. 217-231, 2024, doi: <http://dx.doi.org/10.20473/jisebi.10.2.217-231>.

9. D. Mustikasari, I. Widaningrum, R. Arifin, and W. H. E. Putri, "Comparison of effectiveness of stemming algorithms in indonesian documents," in 2nd Borobudur International Symposium on Science and Technology (BIS-STE 2020), 2021: Atlantis Press, pp. 154-158.

10. M. A. Nq, L. P. Manik, and D. Widiyatmoko, "Stemming Javanese: Another Adaptation of the Nazief-Adriani Algorithm," in 2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), 2020: IEEE, pp. 627-631.

11. A. P. Wibawa and M. N. m. Hakim, "Stemming Bahasa Jawa Menggunakan Damerau Levenshtein Distance (Dld)," *Jurnal Teknik Informatika,* vol. 14, no. 1, pp. 22-27, 2021, doi: 10.15408/jti.v14i1.15010.

12. F. Amin, W. HADIKURNIAWATI, S. WIBISONO, H. FEBRUARIYANTI, and J. S. WIBOWO, "A HYBRID METHOD OF RULE-BASED AND STRING MATCHING STEMMER FOR JAVANESE LANGUAGE," *Journal of Theoretical & Applied Information Technology,* vol. 95, no. 19, 2017.

13. I. A. Ropikoh, R. Abdulhakim, U. Enri, and N. Sulistiyowati, "Penerapan Algoritma Support Vector Machine (SVM) untuk Klasifikasi Berita Hoax Covid-19," *Journal of Applied Informatics and Computing,* vol. 5, no. 1, pp. 64-73, 2021.

14. J. Ahmed and M. Ahmed, "Online news classification using machine learning techniques," IIUM Engineering Journal, vol. 22, no. 2, pp. 210-225, 2021.

15. H. B. Dogru, S. Tilki, A. Jamil, and A. A. Hameed, "Deep learning-based classification of news texts using doc2vec model," in 2021 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA), 2021: IEEE, pp. 91-96.

16. T. S. Sabrila, Y. Azhar, and C. S. K. Aditya, "Analisis Sentimen Tweet Tentang UU Cipta Kerja Menggunakan Algoritma SVM Berbasis PSO," JISKA (Jurnal Informatika Sunan Kalijaga), vol. 7, no. 1, pp. 10-19, 2022.

17. S. Ghosh, A. Dasgupta, and A. Swetapadma, "A study on support vector machine based linear and non-linear pattern classification," in 2019 International conference on intelligent sustainable systems (ICISS), 2019: IEEE, pp. 24-28.

18. L. Triyono, R. Gernowo, M. Rahaman, and T. R. Yudantoro, "Indonesian Fake News Detection Using Various Machine Learning Technique," International Journal on Informatics Visualization, vol. 7, no. 3, pp. 726-732, 2023.

19. N. Kalcheva, M. Karova, and I. Penev, "Comparison of the accuracy of SVM kemel functions in text classification," in 2020 International Conference on Biomedical Innovations and Applications (BIA), 2020: IEEE, pp. 141-145.

20. L. Irham, A. Adiwijaya, and U. Wisesty, "Klasifikasi Berita Bahasa Indonesia Menggunakan Mutual Information dan Support Vector Machine. JURNAL MEDIA INFORMATIKA BUDIDARMA, 3 (4), 284–292," ed, 2019.

21. S. Sudianto, A. D. Sripamuji, I. R. Ramadhanti, R. R. Amalia, J. Saputra, and B. Prihatnowo, "Penerapan Algoritma Support Vector Machine dan Multi-Layer Perceptron pada Klasifikasi Topik Berita," *Jurnal Nasional Pendidikan Teknik Informatika: JANAPATI,* vol. 11, no. 2, pp. 84-91, 2022.

22. A. Fahda and A. Purwarianti, "A statistical and rule-based spelling and grammar checker for Indonesian text," in 2017 international conference on data and software engineering (icodse), 2017: IEEE, pp. 1-6.

23. M. A. P. Subali and C. Fatichah, "KOMBINASI METODE RULE-BASED DAN N-GRAM STEMMING UNTUK MENGENAL STEMMER BAHASA BALI," *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK),* vol. 6, no. 2, 2019.

24. S. Yuliani, S. Sahib, M. Abdollah, Y. Wijaya, and N. Yusoff, "Hoax news validation using similarity algorithms," in Journal of Physics: Conference Series, 2020, vol. 1524, no. 1: IOP Publishing, p. 012035.

25. P. Santoso, P. Yuliawati, R. Shalahuddin, and A. P. Wibawa, "Damerau Levenshtein Distance for Indonesian Spelling Correction," *Jurnal Informatika Ahmad Dahlan,* vol. 13, no. 2, pp. 11-15, 2019.

26. B. V. Indriyono, "Kombinasi damerau levenshtein dan jaro-winkler distance untuk koreksi kata bahasa inggris," Jurnal Teknik Informatika Dan Sistem Informasi, vol. 6, no. 2, 2020.

27. D. Tuhenay, "Perbandingan Klasifikasi Bahasa Menggunakan Metode Naïve Bayes Classifier (NBC) Dan Support Vector Machine (SVM)," JIKO (Jurnal Informatika dan Komputer), vol. 4, no. 2, pp. 105-111, 2021.

28. M. Maryamah, A. Z. Arifin, R. Sarno, and A. M. Hasan, "Adapting Google translate using dictionary and word embedding for Arabic-Indonesian cross-lingual information retrieval," in *2020 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)*, 2021: IEEE, pp. 205-209.

29. Balai Bahasa Provinsi Daerah Istimewa Yogyakarta, "Kamus Bahasa Jawa-Indonesia (KBJI)," Kementerian Pendidikan Dasar dan Menengah, [*Online*]. Tersedia:  [https://kbji.kemendikdasmen.go.id/](https://kbji.kemdikbud.go.id/).

30. L. A. Yates, Z. Aandahl, S. A. Richards, and B. W. Brook, "Cross validation for model selection: a review with examples from ecology," Ecological Monographs, vol. 93, no. 1, p. e1557, 2023.