Enhancing Machine Learning Model Performance in Addressing Class Imbalance

Lucky Lhaura Van FC[1], M. Khairul Anam*[2], Muhammad Bambang Firdaus[3], Yogi Yunefri[4], Nadya Alinda Rahmi[5]

[1,4] Department of Informatics Engineering, Faculty of Computer Science, University of Lancang Kuning, Pekanbaru, Indonesia

[2]Department of Informatics, Faculty of Engineering, University of Samudra, Langsa, Indonesia

[3]Department of Informatics, Faculty of Engineering, University of Mulawarman, Samarinda, Indonesia

[5]Department of Informatics, Faculty of Engineering, University of UPI "YPTK'', Padang, Indonesia

e-mail: [1]lucky@unilak.ac.id., ***[2]khairula210@gmail.com**, [3]bambangf@fkti.unmul.ac.id, [4]yogiyunefri@unilak.ac.id, [5]nadyaalindaa@upiyptk.ac.id

Abstract

This research investigates methods for addressing class imbalance in machine learning models, focusing on the Support Vector Machine (SVM) algorithm. We apply Over-sampling (SMOTE) and Under-sampling techniques to a dataset with class imbalance and evaluate the performance of SVM using these methods. The data consists of Twitter posts related to the 2024 electoral discourse. The findings indicate that incorporating SMOTE effectively enhances the performance of SVM models, particularly within the SVM Polynomial variant. However, the use of Under-sampling shows limited impact on improving SVM model performance. This study provides valuable insights for researchers and practitioners in choosing appropriate strategies for handling class imbalance in machine learning models.

Keywords—Machine Learning, Support Vector Machine, SMOTE, Over-sampling, Under-sampling

1. INTRODUCTION

Class imbalance is one of the challenges researchers face when conducting modeling using Machine Learning [1]. The consequence of class imbalance is that minority classes are often misclassified as major classes [2]. Skewed data distributions in datasets manifest when a single class is disproportionately represented compared to the others [3]. Numerous approaches exist for tackling this quandary, including ADASYN (Adaptive Synthetic Sampling Approach) [4] and SMOTE (Synthetic Minority Over-sampling Technique) [5].

The Synthetic Minority Over-sampling Technique (SMOTE) is a valuable tool for addressing class imbalance in machine learning datasets. By generating synthetic samples for the minority class, SMOTE aids in training more accurate and robust predictive models. While SMOTE offers several benefits, it is essential to consider its limitations and the appropriate parameters to achieve optimal results.[6]. Overall, SMOTE remains a popular and effective method for tackling class imbalance in various machine-learning applications. [7].

SMOTE has been widely adopted by other researchers, as seen in [8] where class resampling using Under-sampling was conducted. The study predicted company bankruptcies using Complement Naïve Bayes. To address this, the researchers utilized the SMOTE method with Under-sampling techniques, resulting in an accuracy improvement of over 2%. Additionally, another study [9] employed SMOTE with Over-sampling techniques. The KNN algorithm was used to test the data, and the use of SMOTE increased K by 9.97%. Furthermore, [10] also utilized

Over-sampling techniques with SMOTE, leading to improved accuracy across all algorithms used.

From several previous articles, two types of SMOTE techniques are identified: Over-sampling and Under-sampling. Under-sampling involves diminishing the volume of data in the predominant class to conform to the quantity present in the underrepresented class [11]. In contrast, Over-sampling is a technique that augments the representation of the minority class to equalize its size with that of the majority class [6]. This research employs both techniques for comparison purposes.

In this investigation, the data comprised Twitter posts related to the 2024 electoral discourse, and the computational framework employed was the Support Vector Machine (SVM), a conventional supervised learning approach frequently applied in classification tasks [12]. SVM determines the optimal hyperplane by maximizing the margin between different classes [13]. A hyperplane serves as a discriminant function used to delineate categories or groups [14]. In this study, three different SVM kernels are used: RBF, Polynomial, and Linear.

Before conducting the modeling process with SVM, the advantages of the employed techniques, SMOTE with Over-sampling and Under-sampling, are determined. This study conducts data preprocessing through a range of techniques, including data purification, case normalization, lexical normalization, tokenization, stopword elimination, and morphological stemming. Labeling the data, this study utilizes a lexicon-based approach. The lexicon-based method operates by first creating an opinion word dictionary (lexicon). Words found in this dictionary are used to identify whether a sentence contains an opinion or not [15]. The created lexicon is then used for automated labeling in this research.

2. RESEARCH METHODS

Figure 1 illustrates the methodology flowchart of this study.

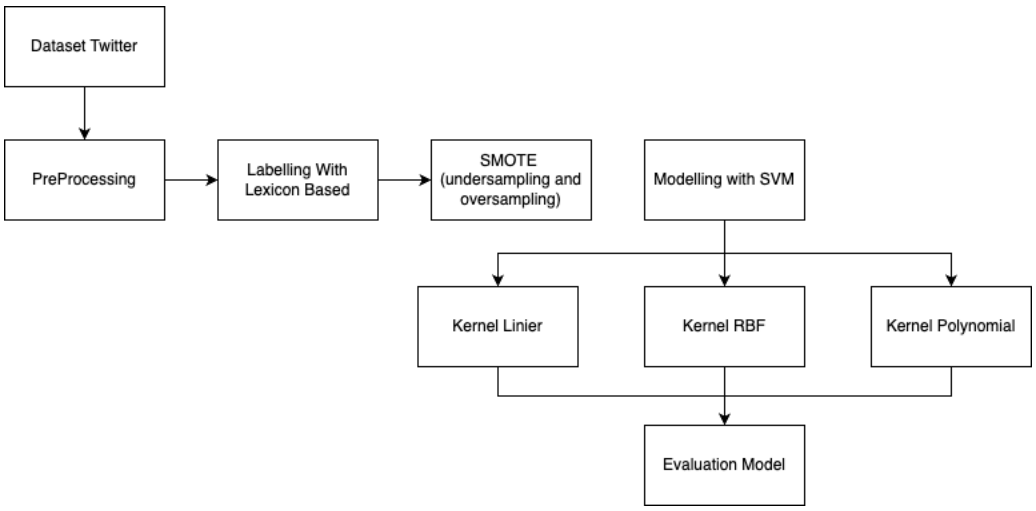

Figure 1. Methodology Flowchart

2.1. Dataset

The dataset for this research is collected from the social media platform Twitter (or X) spanning a period of 3 months, from January to March 2023. The initial dataset comprises 10,001 tweets. The data collection involved using specific keywords related to the research topic. Preprocessing included steps such as removing duplicates, filtering out irrelevant content, and normalizing text, which resulted in data reduction.

2.2. PreProcessing

Data pre-processing denotes the methodological transformation of unrefined data into a more digestible format [16]. This procedure holds paramount significance given that raw data frequently exhibits irregular formatting. Moreover, raw data cannot be processed directly in data mining, making preprocessing essential to facilitate subsequent data analysis [17]. The preprocessing steps are as follows:

a. Data Cleaning

Data preprocessing starts with either data cleaning or filtering [18]. This involves re-selecting raw data and removing incomplete, irrelevant, and inaccurate data. This step helps avoid misunderstandings during data analysis. By applying this process, the quality of the dataset is significantly improved, leading to more accurate and reliable results in subsequent analysis. It ensures that the model is trained on data that is representative and free from noise, thereby enhancing the overall performance and robustness of the machine learning algorithms used in the study.

b. Case Folding

This procedure aims to transform all letters within the document into lowercase [18]. Moreover, symbols such as punctuation marks, numerical digits, and any other non-alphabetic characters are systematically excluded. This is due to their designation as word separators or delimiters, rendering them devoid of any impact on text analysis and interpretation [19]. Furthermore, leading and trailing spaces are removed, a technique known as whitespace removal [20]. By applying these preprocessing steps, the text data becomes standardized and consistent, which significantly enhances the performance and accuracy of the machine learning models. It ensures that the models are not misled by variations in text format and can focus on meaningful content, leading to more reliable and valid analysis outcomes.

c. Normalization

Normalization is a preprocessing method that transforms raw data into a different form to obtain more suitable data for analysis and modeling [21]. This process focuses on scaling the data. Normalizing data helps prevent large initial data ranges from overshadowing smaller data ranges by assigning equal weight to all data points [22]. This is crucial because it ensures that each feature contributes equally to the model's learning process, thereby enhancing the overall performance and accuracy of the model. Without normalization, features with larger scales could dominate the learning process, leading to biased and suboptimal results.

d. Tokenize

The process of tokenization encompasses the segmentation of an assemblage of characters within a given textual composition into individual lexical units [23]. According to [24] Tokenization involves the parsing of an input string by isolating each word. The essence of this procedure lies in the division of each word comprising a document.

e. Stopword Removal

Stopwords are common words that often appear and do not provide significant information and are usually ignored or discarded when creating indices or lists of words [25]. Stopwords are also often considered as noise in text. Stopwords refer to the most common words such as prepositions like "in," "to," "that," and so on.

f. Stemming

Stemming, as a method in natural language processing (NLP), is employed to normalize words by simplifying them to their root or base. This technique seeks to eliminate superfluous

linguistic variations like affixes and verb conjugations, facilitating the identification of common roots across various word forms [26].

2.3. Labeling

After cleaning the data to be processed, the next step is labeling. In this study, labeling is performed using the lexicon-based method. Several studies have stated that using this method can improve accuracy [27], [28]. This study employed a dataset comprising 4800 instances, with a subset of tweets being excluded from analysis during preprocessing due to redundancy.

2.4. Word Weighting With TF-IDF

TF-IDF is an established algorithm employed to assess the significance of a word within a text. This computational process is rooted in both term frequency, which gauges a term's presence in a specific document, and inverse document frequency, which evaluates its prevalence across multiple documents. Through this calculation, TF-IDF is capable of discerning the degree of a word's importance and relevance within a given context [29]. The Term Frequency-Inverse Document Frequency (TF-IDF) method is widely employed for the efficient analysis of voluminous datasets [30]. The TF-IDF algorithm assigns weights to each keyword in each category to find keyword similarities with available categories. Before weighting, five preprocessing steps are performed: sentence segmentation, case folding, tokenizing, filtering, and stemming. Subsequently, TF-IDF weight calculation, query relevance weight, and similarity weight are computed [31].

Based on previous studies discussing the implementation of the TF-IDF method, various formula variations were found in implementing the TF-IDF method for word weighting. The term frequency-inverse document frequency (TF-IDF) metric augments by the occurrences of a term within a document, yet it's counterbalanced by the term's frequency within the overall corpus. The distinct methodologies of TF-IDF weighting schemes frequently serve as the principal mechanism for evaluating and ranking document pertinence as perceived by users. Fundamentally, TF-IDF is an outcome of the multiplication of the term frequency (TF) and the inverse document frequency (IDF). A multitude of approaches exist to ascertain the optimal values for both these extant statistics. In the case of term frequency *tf(t, d)*, the simplest method is to use raw frequency in the document, i.e., how many times term t appears in document d. If we denote raw frequency *t* as *f (t, d)*, the simple *tf* schema is *tf(t, d) = f(t, d)*. The IDF value of a term (word) can be calculated using the following equation:

$$IDF = log_{10}(\frac{D}{dfi}) \quad (1)$$

Where D is the total number of documents containing the term (*t*), and *dfi* is the frequency of occurrence of the word in *D*. The algorithm used to calculate the weight (*W*) of each document for the keyword (query) is:

$$W_{d,t} = tf_{d,t} * IDF_t \quad (2)$$

Information:
d = document number *d*
t = word number *t* from the keyword
W = weight of document *d* for word *t*
tf = term frequency/word frequency

Once the weights (*W*) of each document are known, a sorting process is performed, where the larger the *W* value, the greater the level of similarity of the document to the searched word, and vice versa.

2.5. SMOTE

This study then employs SMOTE to balance classes. Although the data used in this study is nearly balanced, to assess the benefits of SMOTE Under-sampling and Over-sampling are employed. SMOTE works by generating synthetic samples for the minority class, thus ensuring that the classifier is not biased towards the majority class. By balancing the classes, SMOTE helps improve the performance of the model, especially in scenarios where the minority class is of critical interest. The application of both Under-sampling and Over-sampling techniques allows for a comprehensive evaluation of how SMOTE can enhance model accuracy and robustness, providing insights into its effectiveness in different data distributions. This approach not only helps in achieving better model performance but also ensures that the model can generalize well to unseen data, ultimately leading to more reliable predictions.

2.6. SVM

Support Vector Machine (SVM) is a form of supervised learning that is deployed in machine learning to conduct tasks associated with both classification and regression [32]. VM operates by creating a predictive model that delineates data points through a hyperplane, which in turn maximizes the distance, or margin, between the hyperplane and the closest data points [33]. A hyperplane, a mathematical construct typically in multidimensional space, effectively divides a data set into two distinct classes. This division is quantified by the margin, which represents the shortest distance from the hyperplane to the nearest data point in each class. A core objective of support vector machines (SVMs) lies in the task of optimally dividing data into two classes by identifying a hyperplane that maximizes this margin.

Researchers often use SVM as a classifier due to its robustness and effectiveness in high-dimensional spaces. Unlike other machine learning algorithms, SVM is particularly effective when the number of dimensions exceeds the number of samples, and it remains efficient in handling large feature spaces. Furthermore, SVM is versatile due to its use of different kernel functions, allowing it to model complex non-linear relationships. The ability to handle various kernels, such as linear, polynomial, and radial basis function (RBF) kernels, makes SVM a powerful tool for a wide range of classification problems. In this study, SVM was chosen because of its high performance in terms of accuracy and its ability to generalize well to unseen data, making it suitable for the dataset and the specific classification tasks at hand. The kernels used in this study include:

a. SVM Kernel Linear

The basic linear kernel is employed when the data under scrutiny is inherently linearly separable [34]. The linear kernel is suitable when there are many features because mapping to a higher-dimensional space does not significantly improve performance, as in text classification [35]. In text classification, both the number of instances (documents) and the number of features (words) are large [36]. The equation for the linear kernel is:

$$K(x, z) = x^T z \quad (3)$$

Information:
$K(x, z)$: Kernel function that measures the similarity between two input vectors *x* and *z*.
$x^T z$: dot product between vectors *x* and *z*.

This formula effectively measures the linear similarity between two vectors. The higher the value of the dot product, the greater the similarity between the two vectors. Linear kernels are often used because they are simple and efficient to compute and work well when data can be separated linearly in feature space.

b. SVM Kernel RBF

The radial basis function (RBF) kernel is frequently employed in analytical settings where the data does not demonstrate linear separability. Within this context, the RBF kernel features two parameters: gamma and cost [37]. The parameter C, which is part of the support vector machine (SVM) model, is crucial for minimizing misclassification errors in the training data. It defines the trade-off between achieving a low training error and a sufficiently simple decision boundary. On the other hand, the parameter Gamma regulates the influence of individual samples on the decision boundary. Low gamma values indicate that distant points are taken into account when determining the decision boundary, while high gamma values prioritize closer points [38]. The equation for the RBF kernel is:

$$K(x,z) = \exp[-y||x-z||^2] \quad (4)$$

Here is an explanation of the elements in this formula:
$K(x,z)$: The kernel function that measures the similarity between two input vectors x and z.
exp: The exponential function.
y: A scalar parameter (gamma) that determines the width of the Gaussian function. Gamma controls how far the influence of a single sample reaches. A large gamma value means a small radius of influence, resulting in a tighter model. Conversely, a small gamma value means a large radius of influence, resulting in a looser model.
$||x-z||^2$: The squared Euclidean distance between the two vectors x and z.

The RBF kernel function transforms the distance between two data points into a similarity measure ranging from 0 to 1. When two data points are very close to each other, the kernel value approaches 1. Conversely, if two data points are far apart, the kernel value approaches 0. The RBF kernel is highly effective for handling non-linearly separable data by mapping it to a higher-dimensional feature space.

c. SVM Kernel Polynomial

The polynomial kernel constitutes a type of kernel function applied in instances where the data cannot be delineated linearly[39]. The polynomial kernel is especially well-suited for problems in which all training sets have been subjected to the process of normalization [40]. The equation for the polynomial kernel is:

$$K(x,z) = (x^T z)^d \text{ or } (1+x^T z)^d \quad (5)$$

Here is an explanation of the elements in this formula:
$K(x,z)$: The kernel function that measures the similarity between two input vectors x and z.
$x^T z$: The dot product of the vectors x and z.
d: The degree of the polynomial.

For $K(x,z) = (x^T z)^d$: This version of the polynomial kernel calculates the dot product of x and z and then raises it to the power of d. It captures interactions up to the d-th degree, making it suitable for problems where the relationship between the features is polynomial.
For $K(x,z) = (1 + x^T z)^d$: This version includes an additional constant term $(1 + x^T z)$ raised to the power of d. The constant term allows the kernel to account for all polynomial terms up to degree d, including the interaction terms and bias. This version often performs better because it captures a broader range of interactions between the features.

3. RESULT AND DISCUSSION

The following are the modeling results after the pre-processing and word weighting processes. The initial experiment involved SVM modeling without using SMOTE. Table 1 presents the evaluation of the SVM model without SMOTE.

Table 1. SVM Comparison Without SMOTE

Algorithm	Accuracy	Precision	Recall	F1-Score
SVM Linear	87 %	87 %	87 %	87 %
SVM Polynomial	50 %	25 %	50 %	33 %
SVM RBF	83 %	84 %	83 %	83 %

In Table 1, the evaluation results of various performance metrics including Accuracy, Precision, Recall, and F1-Score for SVM without SMOTE are presented for comparison. SVM Linear demonstrates consistent performance across all metrics, with accuracy, precision, recall, and F1-Score all at 87%. This indicates that SVM Linear effectively classifies the data by achieving high accuracy in predicting both positive and negative classes. The superior performance of SVM Linear may be due to the simplicity of the linear hyperplane, which is easier to optimize in less complex feature space.

SVM Polynomial shows significantly lower performance compared to SVM Linear. With an accuracy of only 50%, precision, recall, and F1-Score of 25%, 50%, and 33% respectively, SVM Polynomial struggles to classify the data, resulting in overall poor performance. SVM Polynomial may experience overfitting to the training data, leading to poor performance on the test data.

SVM RBF shows relatively good performance, with an accuracy of 83%. SVM RBF maintains high precision, recall, and F1-Score values, indicating its effectiveness in predicting both positive and negative classes. However, its performance is slightly lower than SVM Linear. The lower performance of SVM RBF compared to SVM Linear could be due to the complexity of the RBF kernel, which may not fully match the current data distribution.

Next, SVM was tested using SVM with Over-sampling. Table 2 presents the evaluation of the SVM model with SMOTE (Over-sampling).

Table 2. SVM Comparison with SMOTE (Over-sampling)

Algorithm	Accuracy	Precision	Recall	F1-Score
SVM Linear	87 %	87 %	87 %	87 %
SVM Polynomial	50 %	25 %	50 %	33 %
SVM RBF	81 %	82 %	81 %	80 %

The evaluation results for the Support Vector Machine (SVM) model with the application of the SMOTE (Synthetic Minority Over-sampling Technique) Over-sampling technique are presented in Table 2. From the table, it can be seen that the use of SMOTE has significantly improved the performance of the SVM model, particularly in SVM polynomials. In SVM Linear, although there is a performance increase, the difference is not as large as in SVM Polynomial and SVM RBF. SVM Linear with SMOTE shows a substantial increase in accuracy from 87% to 91%. It appears that the application of the Over-sampling method has contributed to enhancing the model's capability in classifying data. Additionally, there is a noticeable improvement in precision, recall, and F1-Score metrics, indicating an increased effectiveness of the model in distinguishing between positive and negative classes.

SVM Polynomial also shows significant improvement after using SMOTE, although its accuracy is still below SVM Linear and SVM RBF. The precision, recall, and F1-Score metrics show substantial increases, indicating an improved ability of the model in classifying data. Meanwhile, SVM RBF shows a smaller improvement compared to SVM Linear and Polynomial after the application of SMOTE. However, the improvement is still present in accuracy, precision,

recall, and F1-Score, indicating that the Over-sampling technique still has a positive impact on the model's performance. Overall, the evaluation results show that the use of SMOTE effectively improves the performance of the SVM model in classifying data, particularly in addressing class imbalance.

The third experiment involved SVM using SVM Under-sampling. Table 3 presents the evaluation of the SVM model with SMOTE (Under-sampling).

Table 3. SVM Comparison with SMOTE (Under-sampling)

Algorithm	Accuracy	Precision	Recall	F1-Score
SVM Linear	87 %	88 %	87 %	87 %
SVM Polynomial	50 %	25 %	50 %	33 %
SVM RBF	80 %	82 %	80 %	80 %

In Table 3, the performance results of the Support Vector Machine (SVM) model with the application of SMOTE (Synthetic Minority Over-sampling Technique) combined with Under-sampling are presented. From the table, it can be seen that the use of SMOTE with Under-sampling has a limited impact on improving the performance of the SVM model. SVM Linear shows consistent results with the use of SMOTE Under-sampling, with no significant changes in accuracy, precision, recall, and F1-Score. Despite the application of the Under-sampling technique to address class imbalance, the results remain similar to SVM without SMOTE.

SVM Polynomial also shows a similar pattern with no significant changes in model performance after the application of SMOTE Under-sampling. The accuracy, precision, recall, and F1-Score remain low, indicating that this technique does not provide significant improvement in the model's ability to classify data. SVM RBF shows a slight improvement in some evaluation metrics after the application of SMOTE Under-sampling. However, the improvement is not significant and is still far from the expected performance. This indicates that the combination of SMOTE and Under-sampling techniques does not provide significant improvement in the SVM model's performance when applied to the current dataset. Overall, the evaluation results show that the use of SMOTE with the Under-sampling technique has a limited impact on improving the SVM model's performance. Although intended to address class imbalance, this technique fails to provide significant enhancement in model performance on the tested dataset.

The results obtained in this study can be considered superior compared to previous research. Table 4 provides a comparison with previous studies using different algorithms.

Table 4. Comparison with Previous Research

Researcher	Algorithm	Dataset	Accuracy
[41]	Logistic Regression	Blibli	73 %
[42]	Naïve Bayes Classifier	Play Store	80 %
[43]	K-Nearest Neighbour	Movie Review	86 %
[44]	Decision Tree	Twitter	86 %
[45]	CNN	Twitter	78 %
[46]	Simple Neural Network	Twitter	85 %
This Research	Support Vector Machine	Twitter	87 %

The analysis of the research results indicates that the use of Support Vector Machine (SVM) on the Twitter dataset achieved the highest accuracy of 87%. Compared to previous studies, SVM demonstrated a significant performance improvement. Research using algorithms such as Logistic Regression on the Blibli dataset achieved an accuracy of 73%, and Naïve Bayes Classifier on the Play Store dataset achieved 80%. K-Nearest Neighbour on the Movie Review dataset and Decision Tree on the Twitter dataset each achieved an accuracy of 86%, while CNN on the Twitter dataset only reached 78%. Research using a Simple Neural Network on the Twitter dataset achieved an accuracy of 85%. Therefore, it can be concluded that SVM outperformed

other algorithms used in previous studies on the Twitter dataset, demonstrating its effectiveness in text classification on this platform.

4. CONCLUSION

This study emphasizes the importance of addressing class imbalance in machine learning models, particularly with Support Vector Machine (SVM) algorithms. Evaluations using over-sampling (SMOTE) and under-sampling methods were conducted on datasets with class imbalance. Results indicate that the use of SMOTE significantly enhances SVM model performance, especially for SVM Polynomial, by correcting class imbalance and improving accuracy, precision, recall, and F1-Score. However, integrating SMOTE with under-sampling showed minimal impact on SVM efficiency, failing to significantly improve model performance. These findings highlight the necessity of carefully selecting appropriate class imbalance handling methods tailored to specific dataset characteristics. This study provides valuable insights for researchers and practitioners in choosing effective methodologies for managing class imbalance to enhance machine learning model effectiveness.

For future research, it is recommended to explore combinations of other machine learning algorithms that can enhance the accuracy and stability of text classification models. Additionally, the application of more complex ensemble learning methods can be investigated to address challenges in more intricate sentiment analysis. The implementation of new techniques in data pre-processing can also be explored to ensure that the data used is clean and relevant, thereby improving the overall performance of the models.

5. ACKNOWLEDGMENTS

The author would like to express gratitude to the University of Lancang Kuning for providing financial support for this research.

REFERENCES

[1] J. Tanha, Y. Abdi, N. Samadi, N. Razzaghi, and M. Asadpour, “Boosting methods for multi-class imbalanced data classification: an experimental review,” *J Big Data*, vol. 7, no. 1, Dec. 2020, doi: 10.1186/s40537-020-00349-y.

[2] Y. Pristyanto, “Penerapan Metode Ensemble Untuk Meningkatkan Kinerja Algoritme Klasifikasi Pada Imbalanced Dataset,” *Jurnal TEKNOINFO*, vol. 13, no. 1, pp. 11–16, 2019, doi: 10.33365/jti.v13i1.184.

[3] R. Dwi Fitriani, H. Yasin, D. Statistika, and F. Sains dan Matematika, “Penanganan Klasifikasi Kelas Data Tidak Seimbang Dengan Random Oversampling Pada Naive Bayes (Studi Kasus: Status Peserta KB IUD di Kabupaten Kendal),” *JURNAL GAUSSIAN*, vol. 10, no. 1, pp. 11–20, 2021, doi: 10.14710/j.gauss.10.1.11-20.

[4] N. G. Ramadhan, “Comparative Analysis of ADASYN-SVM and SMOTE-SVM Methods on the Detection of Type 2 Diabetes Mellitus,” *Scientific Journal of Informatics*, vol. 8, no. 2, pp. 276–282, Nov. 2021, doi: 10.15294/sji.v8i2.32484.

[5] M. K. Anam, T. A. Fitri, Agustin, Lusiana, M. B. Firdaus, and A. T. Nurhuda, “Sentiment Analysis for Online Learning using The Lexicon-Based Method and The Support Vector Machine Algorithm,” *ILKOM Jurnal Ilmiah*, vol. 15, no. 2, pp. 290–302, 2023, doi: 10.33096/ilkom.v15i2.1590.290-302.

[6] C. Kaope and Y. Pristyanto, "The Effect of Class Imbalance Handling on Datasets Toward Classification Algorithm Performance," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 22, no. 2, pp. 227–238, Mar. 2023, doi: 10.30812/matrik.v22i2.2515.

[7] D. Elreedy, A. F. Atiya, and F. Kamalov, "A theoretical distribution analysis of synthetic minority oversampling technique (SMOTE) for imbalanced learning," *Mach Learn*, 2023, doi: 10.1007/s10994-022-06296-4.

[8] W. I. Sabilla and C. B. Vista, "Implementasi SMOTE dan Under Sampling pada Imbalanced Dataset untuk Prediksi Kebangkrutan Perusahaan," *Jurnal Komputer Terapan*, vol. 7, no. 2, pp. 329–339, 2021, doi: 10.35143/jkt.v7i2.5027.

[9] F. Dwi Astuti and F. Nova Lenti, "Implementasi SMOTE untuk mengatasi Imbalance Class pada Klasifikasi Car Evolution menggunakan K-NN," *Jurnal JUPITER*, vol. 13, no. 1, pp. 89–98, 2021.

[10] A. Syukron, E. Saputro, Sardiarinto, and P. Widodo, "Penerapan Metode Smote Untuk Mengatasi Ketidakseimbangan Kelas Pada Prediksi Gagal Jantung," *Jurnal Teknologi Informasi dan Terapan (J-TIT)*, vol. 10, no. 1, pp. 2580–2291, 2023, doi: 10/25047/jtit.v10i1.312.

[11] B. Santoso, H. Wijayanto, K. A. Notodiputro, and B. Sartono, "Synthetic over Sampling Methods for Handling Class Imbalanced Problems : A Review," in *IOP Conference Series: Earth and Environmental Science*, Institute of Physics Publishing, Apr. 2017. doi: 10.1088/1755-1315/58/1/012031.

[12] E. M. S. Rochman *et al.*, "Classification of Thesis Topics Based on Informatics Science Using SVM," in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, May 2021, pp. 1–7. doi: 10.1088/1757-899x/1125/1/012033.

[13] Saikin, S. Fadli, and M. Ashari, "Optimization of Support Vector Machine Method Using Feature Selection to Improve Classification Results," *JISA (Jurnal Informatika dan Sains)*, vol. 4, no. 1, pp. 22–27, 2021, doi: 10.31326/jisa.v4i1.881.

[14] V. V., R. A. C, R. Mohammed, S. K. V, and P. S. Kumthekar, "Support Vector Machine Implementation to Separate Linear and Non-Linear Dataset," *Saudi Journal of Engineering and Technology*, vol. 8, no. 1, pp. 4–15, Jan. 2023, doi: 10.36348/sjet.2023.v08i01.002.

[15] R. H. Muhammadi, T. G. Laksana, and A. B. Arifa, "Combination of Support Vector Machine and Lexicon-Based Algorithm in Twitter Sentiment Analysis," *KHAZANAH INFORMATIKA*, vol. 8, no. 1, pp. 59–71, 2022, doi: 10.23917/khif.v8i1.15213.

[16] A. N. Ulfah and M. K. Anam, "Analisis Sentimen Hate Speech Pada Portal Berita Online Menggunakan Support Vector Machine (SVM)," *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, vol. 7, no. 1, pp. 1–10, 2020, doi: 10.35957/jatisi.v7i1.196.

[17] M. A. Jassim and S. N. Abdulwahid, "Data Mining preparation: Process, Techniques and Major Issues in Data Analysis," in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, Mar. 2021, p. 012053. doi: 10.1088/1757-899x/1090/1/012053.

[18] R. Rudiman and N. A. Rahmi, "Latent Dirichlet Allocation Utilization as a Text Mining Method to Elaborate Learning Effectiveness," *JSE Journal of Science and Engineering*, vol. 1, no. 1, pp. 23–29, Sep. 2023, doi: 10.30650/jse.v1i1.3680.

[19] H. Mukhtar, J. Al Amien, and M. A. Rucyat, "Filtering Spam Email menggunakan Algoritma Naïve Bayes," *Jurnal CoSciTech (Computer Science and Information Technology)*, vol. 3, no. 1, pp. 9–19, May 2022, doi: 10.37859/coscitech.v3i1.3652.

[20] M. A. Fauzi, "Word2Vec model for sentiment analysis of product reviews in Indonesian language," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 1, pp. 525–530, Feb. 2019, doi: 10.11591/ijece.v9i1.pp525-530.

[21] K. M. G. S. Karunarathna and R. A. H. M. Rupasingha, "Learning to Use Normalization Techniques for Preprocessing and Classification of Text Documents," *International Journal of Multidisciplinary Studies (IJMS)*, vol. 9, pp. 69–81, 2022, doi: 10.31357/ijms.v9i2.

[22] D. Singh and B. Singh, "Investigating the impact of data normalization on classification performance," *Appl Soft Comput*, vol. 97, pp. 1–23, Dec. 2020, doi: 10.1016/j.asoc.2019.105524.

[23] M. K. Anam, M. I. Mahendra, W. Agustin, Rahmaddeni, and Nurjayadi, "Framework for Analyzing Netizen Opinions on BPJS Using Sentiment Analysis and Social Network Analysis (SNA)," *Intensif*, vol. 6, no. 1, pp. 2549–6824, 2022, doi: 10.29407/intensif.v6i1.15870.

[24] K. Davagdorj, L. Wang, M. Li, V. H. Pham, K. H. Ryu, and N. Theera-Umpon, "Discovering Thematically Coherent Biomedical Documents Using Contextualized Bidirectional Encoder Representations from Transformers-Based Clustering," *Int J Environ Res Public Health*, vol. 19, no. 10, pp. 1–21, May 2022, doi: 10.3390/ijerph19105893.

[25] S. Sarica and J. Luo, "Stopwords in technical language processing," *PLoS One*, vol. 16, no. 8, pp. 1–13, Aug. 2021, doi: 10.1371/journal.pone.0254937.

[26] P. Koirala and A. Shakya, "A Nepali Rule Based Stemmer and its performance on different NLP applications," *ArXiv*, 2020, doi: 10.48550/arXiv.2002.09901.

[27] A. Syakur, "Implementasi Metode Lexicon Base Untuk Analisis Sentimen Kebijakan Pemerintah Dalam Pencegahan Penyebaran Virus Corona Covid-19 Pada Twitter," *Jurnal Ilmiah Informatika Komputer*, vol. 26, no. 3, pp. 247–260, 2021, doi: 10.35760/ik.2021.v26i3.4720.

[28] F. Amaliah, I. Kadek, and D. Nuryana, "Perbandingan Akurasi Metode Lexicon Based Dan Naive Bayes Classifier Pada Analisis Sentimen Pendapat Masyarakat Terhadap Aplikasi Investasi Pada Media Twitter," *Journal of Informatics and Computer Science*, vol. 3, no. 3, pp. 384–393, 2022, doi: 10.26740/jinacs.v3n03.p384-393.

[29] D. Septiani and I. Isabela, "Analisis Term Frequency Inverse Document Frequency (Tf-Idf) Dalam Temu Kembali Informasi Pada Dokumen Teks," *SINTESIA: Jurnal Sistem dan Teknologi Informasi Indonesia*, vol. 1, no. 2, pp. 81–88, 2022.

[30] H. Fan and Y. Qin, "Research on Text Classification Based on Improved TF-IDF Algorithm," in *International Conference on Network, Communication, Computer Engineering*, 2018, pp. 501–506. doi: 10.2991/ncce-18.2018.79.

[31] C.-Z. Liu, Y.-X. Sheng, Z.-Q. Wei, and Y.-Q. Yang, "Research of Text Classification Based on Improved TF-IDF Algorithm," in *International Conference of Intelligent Robotic and Control Engineering*, 2018, pp. 218–222. doi: 10.1109/IRCE.2018.8492945.

[32] R. Thiruvengatanadhan, "Music Classification using MFCC and SVM," *International Research Journal of Engineering and Technology (IRJET)*, vol. 5, no. 9, pp. 922–924, 2018.

[33] J. Cao, G. Lv, C. Chang, and H. Li, "A Feature Selection Based Serial SVM Ensemble Classifier," *IEEE Access*, vol. 7, pp. 144516–144523, 2019, doi: 10.1109/ACCESS.2019.2917310.

[34] B. Yassin, C. Mohamed, and Y. Al-Amrani, "A Nonlinear Support Vector Machine Analysis Using Kernel Functions for Nature and Medicine," in *E3S Web of Conferences*, EDP Sciences, Nov. 2021, pp. 1–5. doi: 10.1051/e3sconf/202131901103.

[35] M. Awad and R. Khanna, "Support Vector Machines for Classification," in *Efficient Learning Machines Theories, Concepts, and Applications For Engineers and System Designers*, Apress Media, 2015, pp. 39–66.

[36] S. Al Qodrin, N. Yusliani, and A. Syahrini, "Classification of Indonesian Questions Using the Support Vector Machine Algorithm and Mutual Information Feature Selection," *Jurnal JUPITER*, vol. 14, no. 2, p. 44, 2022, doi: 10.5281./4796/5.jupiter.2022.10.

[37] S. H. Hasanah, "Classification Support Vector Machine In Breast Cancer Patients," *BAREKENG: Jurnal Ilmu Matematika dan Terapan*, vol. 16, no. 1, pp. 129–136, Mar. 2022, doi: 10.30598/barekengvol16iss1pp129-136.

[38] A. A. Ewees, A. A. Hemedan, A. E. Hassanien, and A. T. Sahlol, "Optimized support vector machines for unveiling mortality incidence in Tilapia fish," *Ain Shams Engineering Journal*, vol. 12, no. 3, pp. 3081–3090, Sep. 2021, doi: 10.1016/j.asej.2021.01.014.

[39] R. Mukarramah, D. Atmajaya, and L. B. Ilmawan, "Performance comparison of support vector machine (SVM) with linear kernel and polynomial kernel for multiclass sentiment analysis on twitter," *ILKOM Jurnal Ilmiah*, vol. 13, no. 2, pp. 168–174, 2021, doi: 10.33096/ilkom.v13i2.851.168-174.

[40] F. R. Lumbanraja, R. A. Saputra, K. Muludi, A. Hijriani Dan, and A. Junaidi, "Implementasi Support Vector Machine dalam Memprediksi Harga Rumah pada Perumahan di Kota Bandar Lampung," *Jurnal Pepadun*, vol. 2, no. 3, pp. 327–335, 2021, doi: 10.23960/pepadun.v2i3.90.

[41] S. A. H. Bahtiar, C. K. Dewa, and A. Luthfi, "Comparison of Naïve Bayes and Logistic Regression in Sentiment Analysis on Marketplace Reviews Using Rating-Based Labeling," *Journal of Information Systems and Informatics*, vol. 5, no. 3, pp. 915–927, Aug. 2023, doi: 10.51519/journalisi.v5i3.539.

[42] R. A. Sitorus and I. Zufria, "Application of the Naïve Bayes Algorithm in Sentiment Analysis of Using the Shopee Application on the Play Store," *Digital Zone*, vol. 15, no. 1, pp. 53–65, 2024, doi: 10.31849/digitalzone.v15i1.19828.

[43] I. Prayoga, M. Dwifebri p, and Adiwijaya, "Sentiment Analysis on Indonesian Movie Review Using KNN Method With the Implementation of Chi-Square Feature Selection," *Jurnal Media Informatika Budidarma*, vol. 7, no. 1, pp. 369–375, 2023, doi: 10.30865/mib.v7i1.5522.

[44] A. Angdresey and G. Saroinsong, "The Decision Tree Algorithm on Sentiment Analysis: Russia and Ukraine War," *Jurnal Sisfotenika*, vol. 13, no. 2, pp. 192–200, 2023, doi: 10.30700/jst.v13i2.1397.

[45] R. A. Rudiyanto and E. B. Setiawan, "Sentiment Analysis Using Convolutional Neural Network (CNN) and Particle Swarm Optimization on Twitter," *JITK (Jurnal Ilmu Pengetahuan dan Teknologi Komputer)*, vol. 9, no. 2, pp. 188–195, Feb. 2024, doi: 10.33480/jitk.v9i2.5201.

[46] A. C. M. V. Srinivas, Ch. Satyanarayana, Ch. Divakar, and K. P. Sirisha, "Sentiment Analysis using Neural Network and LSTM," in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, Feb. 2021, pp. 1–7. doi: 10.1088/1757-899x/1074/1/012007.