ECTI Transactions on Computer and Information Technology
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

A Machine Learning Approach for Multi-Label Classification in Candidate Election Social Media Analysis

Herman Yuliansyah[1], Ricy Ardiansyah[2] and Anton Yudhana[3]

ABSTRACT

Multi-label text classification in social media comments presents a significant challenge in natural language processing. Several previous studies have conducted sentiment analysis on candidate (presidential and gubernatorial) elections using machine learning approaches. However, an opinion can contain more than one category or label simultaneously, such as sentiment, candidate, or certain issues. This study proposes a multi-label classification model to improve accuracy, addressing challenges such as complex language structure, non-standard word usage, and imbalanced data. The proposed model is compared with three popular classification algorithms: Naive Bayes (NB), Support Vector Machine (SVM), and K-Nearest Neighbours (KNN), for handling multi-label text classification tasks. The proposed model comprises a classification pipeline that includes data preprocessing, feature extraction using TF-IDF, and the integration of the GridSearchCV technique to enhance algorithm performance and effectiveness. The evaluation is conducted using multi-label metrics such as Precision, Recall, and F1-Score. The experiment results showed that SVM with GridSearchCV provided the best performance in terms of precision and generalization on the gubernatorial election dataset. SVM + GridSearchCV yielded scores of 97.4% and 99.2% for candidate labels, and 99.2% and 99.0% for sentiment labels. While NB and KNN also showed improvements, their performance was not as significant as SVM. NB outperformed in computational performance, whereas KNN demonstrated poor performance on high-dimensional data.

Article information:

Keywords: Aspect-based Sentiment Analysis, Algorithm Comparison, Candidate Election, Imbalanced Data, Natural Language Processing

Article history:
Received: September 5, 2025
Revised: November 27, 2025
Accepted: November 27, 2025
Published: January 3, 2026 (Online)

DOI: 10.37936/ecti-cit.2026201.263596

1. INTRODUCTION

The advancement of information and communication technology has driven the rapid growth of social media as the main platform for people to express their opinions, criticisms, and support for various issues [1]. Comments published on social media, such as Twitter, Facebook, and Instagram, not only reflect individual expressions but can also be a rich source of data for public opinion analysis and data-based decision making. However, the natural characteristics of social media comment data are unstructured, diverse, and often contain more than one topic or sentiment, especially in regional head elections [2]. This poses challenges in the analysis process, especially in the context of multi-label text classification.

In multi-label text classification, one comment can be categorized into more than one class or label at the same time. For example, a comment can contain negative sentiment as well as be related to political issues from other candidates [3]. The main challenge in this multi-label classification lies not only in the complexity of the relationship between labels, but also in the imbalance in the distribution of labels, where some labels appear very dominant while others are only a few [4].

[1]The author is with the Department of Informatics, Universitas Ahmad Dahlan, Yogyakarta, Indonesia, Email: herman.yuliansyah@tif.uad.ac.id
[2]The author is with the Master Program of Informatics, Universitas Ahmad Dahlan, Yogyakarta, Indonesia, Email: ardianriki199@gmail.com
[3]The author is with the Department of Electrical Engineering, Universitas Ahmad Dahlan, Yogyakarta, Indonesia, Email: eyudhana@ee.uad.ac.id
[1]Corresponding author: herman.yuliansyah@tif.uad.ac.id

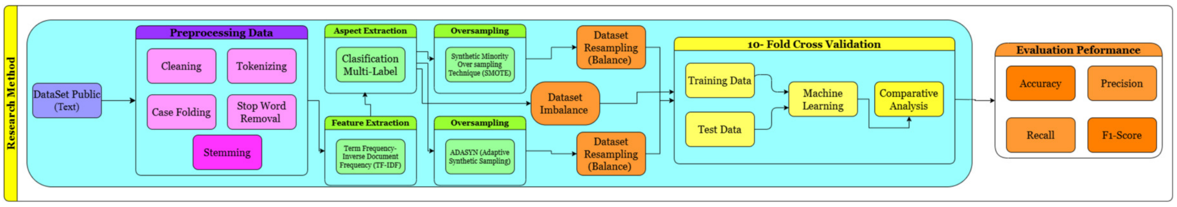


Fig.1: *Proposed machine learning pipeline.*

Various machine learning algorithms have been used to handle multi-label text classification, including Naive Bayes (NB), Support Vector Machine (SVM), and K-Nearest Neighbor (KNN). NB is known to have fast and simple performance [5], but is often not accurate enough for complex data. SVM excels in separating classes with optimal margins and is suitable for high-dimensional data, while KNN offers an easy-to-understand data similarity-based approach. However, the performance of these three methods can be significantly affected if the label distribution in the dataset is very imbalanced and optimization of the three methods using the addition of the GridSearchCV technique to improve the performance and effectiveness of the algorithm combination [6].

To overcome the problem of data imbalance, oversampling approaches are used, such as Synthetic Minority Over-sampling Technique (SMOTE) and Adaptive Synthetic Sampling (ADASYN) [7]. SMOTE works by creating synthetic examples of the minority class to balance the data distribution, while ADASYN is a development of SMOTE that adjusts the amount of synthesis based on the learning difficulty of the minority data. The use of these two methods aims to improve the algorithm's ability to improve overall classification performance, especially in the multi-label context [8].

This study aims to propose and compare the performance of NB, SVM, and KNN algorithms in multi-label text classification on social media comment data, both before and after the application of SMOTE and ADASYN. The evaluation was conducted using accuracy, precision, recall, and F1-score metrics to assess the effectiveness of the combination of algorithms and data balancing techniques in handling the complexity and label imbalance in social media comment data for the Gubernatorial Election. The results of this study are expected to contribute to the selection of optimal methods for automatic public opinion classification in various digital social contexts.

2. METHOD

Based on Figure 1, the process started by loading the dataset. An opinion text serves as the input variable and six output classes, namely the number of three presidential candidates and three gubernatorial candidates. These input variables are transformed into numeric data using TF-IDF. Furthermore, the SMOTE and ADASYN methods are used to handle data imbalance in each algorithm and compare the results of several methods with performance optimization using GridSearchCV and to classify the presidential and gubernatorial candidate data sets. Cross-validation is used to calculate the predicted performance values of the Naive Bayes, SVM, and KNN algorithms.

2.1 Data Collection

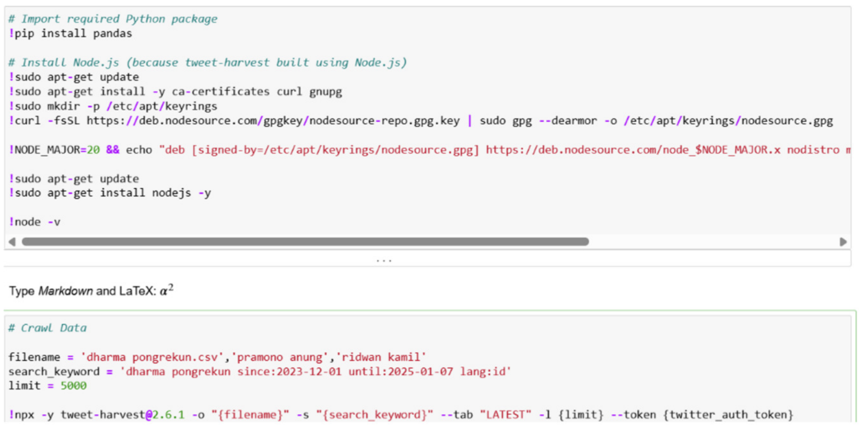
This study uses two data sets, where the first data set is the President and the second data set is the Governor. The first dataset uses a `https://data.mendeley.com/datasets/7w5zvr8jgp/5`. The second dataset was collected via Twitter using the Tweet-Harvest method, which utilizes Twitter API authentication tokens to retrieve tweets and user metadata [9]. The majority of user tweets are in English because more analysis tools are available [10]. Data was collected using a Python module scraping [11]. The scraped datasets were stored in CSV format

Table 1: *Presidential Candidate Dataset.*

Candidate Label Capres	Number of Data	Positive Labels	Negative Labels
Anies Baswedan	10001	6455	3546
Prabowo Subianto	10002	7369	2633
Ganjar Pranowo	10002	7831	2171

Table 2: *Gubernatorial Candidate Dataset.*

Candidate Label Cagub	Number of Data	Positive Labels	Negative Labels
Pramono Anung	4048	2945	1103
Ridwan Kamil	4026	1790	2236
Dharma Pongrekun	4024	1919	2105



```
# Import required Python package
!pip install pandas

# Install Node.js (because tweet-harvest built using Node.js)
!sudo apt-get update
!sudo apt-get install -y ca-certificates curl gnupg
!sudo mkdir -p /etc/apt/keyrings
!curl -fsSL https://deb.nodesource.com/gpgkey/nodesource-repo.gpg.key | sudo gpg --dearmor -o /etc/apt/keyrings/nodesource.gpg

!NODE_MAJOR=20 && echo "deb [signed-by=/etc/apt/keyrings/nodesource.gpg] https://deb.nodesource.com/node_$NODE_MAJOR.x nodistro m

!sudo apt-get update
!sudo apt-get install nodejs -y

!node -v
```

Type *Markdown* and LaTeX: α^2

```
# Crawl Data

filename = 'dharma pongrekun.csv','pramono anung','ridwan kamil'
search_keyword = 'dharma pongrekun since:2023-12-01 until:2025-01-07 lang:id'
limit = 5000

!npx -y tweet-harvest@2.6.1 -o "{filename}" -s "{search_keyword}" --tab "LATEST" -l {limit} --token {twitter_auth_token}
```

Fig.2: *Data collection.*

to facilitate further analysis. The first and second datasets are shown in Tables 1 and 2.

Figure 2 shows the program code to collect data from the 2024 Jakarta gubernatorial candidates, using the Twitter API with keywords on the three candidates, and the data time range from the end of 2023 to the beginning of 2025 was collected, along with the limit on the number of tweets retrieved per day.

2.2 Data Preprocessing

Text preprocessing techniques play an important role in converting unstructured textual data into structured formats, which facilitate analysis and prediction for various tasks, such as sentiment analysis. Text preprocessing is conducted with the aim that the initial data is processed through several stages until the data is completely ready to be used [12]. Text cleaning is the process of removing unwanted words to reduce interference in the classification process. Words that are removed, for example, characters [13]. Case folding is to return all words to all lowercase letters so that the processed text data is all in the same form. Tokenization is a method of roughly dividing a series of characters in text into words to distinguish between certain characters that may or may not be treated as word breaks [14]. Then the process of removing stopwords is carried out, namely the filtering process, selecting important words from the token results, namely what words are used to represent documents [15]. Later, stemming is conducted for the process of mapping and parsing various forms of words into their basic forms [16]. Figure 3 explains the pre-processing data flow for the Candidate for Governor dataset, which is part of the research process to obtain clean data, with the aim of facilitating further analysis.

2.3 Feature Weighting

TF-IDF is one of the earliest and most common unsupervised weighting methods [17]. The intuition behind TF-IDF is that, for some contexts, some terms are more important than others to describe a document. For example, a term that appears in all documents has no substantial relevance to help identify documents. Equation 1 describes TF-IDF, where N is the number of documents in the corpus and DF (ti) corresponds to the frequency of documents in which term ti appears in the collection. TF-IDF and TF are considered unsupervised term weighting schemes because they do not take into account class information [18].

$$W_{TF.IDF(T_i)} = TF(t_i, d_j)x\log\left(\frac{D}{DF(t_i)}\right) \tag{1}$$

2.4 Aspect Based Sentiment Analysis

Aspect-Based Sentiment Analysis (ABSA), as a multi-label classification, is a branch of sentiment analysis that is more specific and in-depth than the traditional sentiment approach. If sentiment analysis

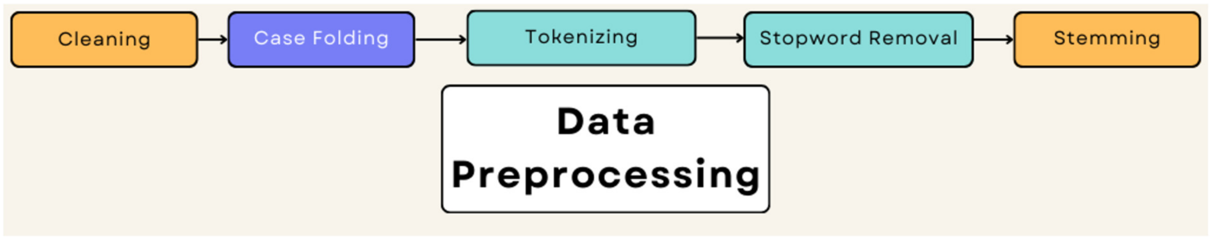


Fig.3: *Data preprocessing.*

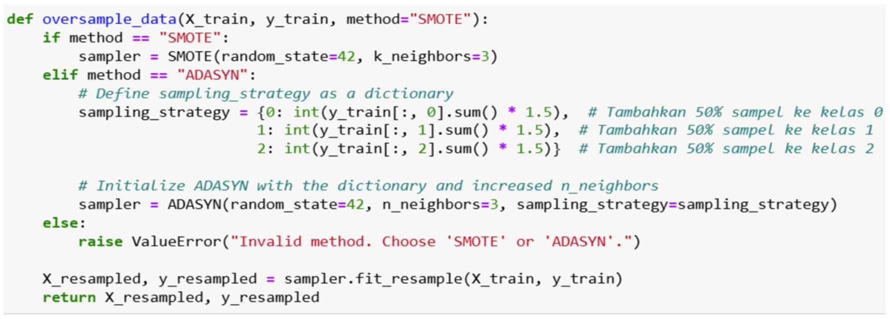
(single-label classification) only focuses on determining the overall polarity of an opinion (positive, negative, or neutral), then ABSA aims to identify and analyze sentiments directed at specific aspects of an entity in the text [19]. Overall, Figure 4 this code snippet is part of an important step in ABSA, which is to convert data in text format and label lists into a numeric format that can be processed by machine learning algorithms such as Naive Bayes or SVM. This label binarization process allows the system to perform multi-aspect classification simultaneously, for example, identifying public sentiment towards various political candidates or the various issues they support. In the context of politics, this process supports Opinion Target Extraction (OTE) to detect who is the target of the opinion, Aspect Category Detection (ACD) to determine the category of aspects, such as candidates or issues, and Sentiment Polarity (SP) to classify public sentiment towards these aspects [20].

2.5 Synthetic Minority Over-sampling Technique

The SMOTE algorithm is used to generate new observations. SMOTE is an over-sampling technique where, instead of duplication, minority class observations are over-sampled by generating synthetic observations [21]. Each minority class sample is over-sampled by generating synthetic samples along the line segments connecting all or some of the K nearest neighbors of the minority class [22]. After that, the minority and majority class observations are merged. Figure 4 explains the oversample_data function to improve model performance on candidate labels for imbalanced data in SMOTE and ADASYN. Synthetic data is created based on nearest neighbors with the parameter k_neighbors = 3. (x_train), label (y_train), and the oversampling method. This function receives input feature data. ADASYN uses additional synthetic data adaptively as much as 50% of the number of each class, based on the sampling strategy for candidate labels. The final result of this function is new training data (x_resampled, y_resampled) that is already balanced.

2.6 Adaptive Synthetic Sampling

Adaptive synthetic sampling, often known as ADASYN, is an additional oversampling method used in Imlearn (Imbalanced Learn). ADASYN is almost similar to SMOTE, with only one important difference [23]. The ADASYN technique is based on the adaptive generation of minority data samples according to their distribution. By assessing the learning challenge of minority samples, ADASYN oversampling technology aims to estimate the weighted distribution of minority samples [24]. Figure 5 explains the oversample_data function to improve model performance on sentiment labels for imbalanced data on SMOTE and ADASYN. There is a change for the ADASYN model, additional synthetics of 50% of the number of each class are not used on the sentiment label, because minority samples can be used. Based on the sampling for synthetic data, sentiment labels are made based on the nearest neighbors with the parameter k_neighbors = 3. (X_train), label (y_train),

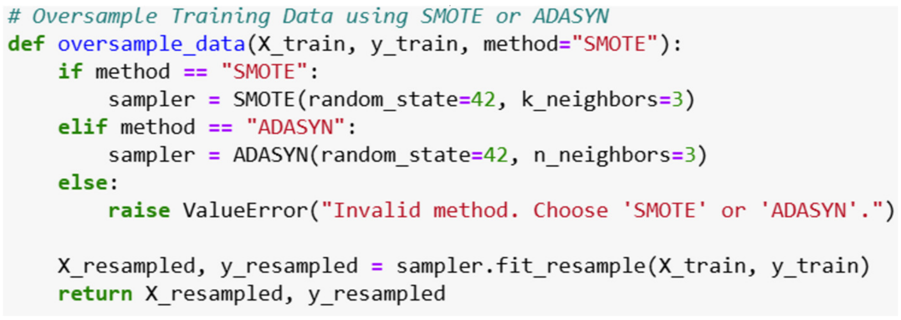


```
def oversample_data(X_train, y_train, method="SMOTE"):
    if method == "SMOTE":
        sampler = SMOTE(random_state=42, k_neighbors=3)
    elif method == "ADASYN":
        # Define sampling_strategy as a dictionary
        sampling_strategy = {0: int(y_train[:, 0].sum() * 1.5),  # Tambahkan 50% sampel ke kelas 0
                             1: int(y_train[:, 1].sum() * 1.5),  # Tambahkan 50% sampel ke kelas 1
                             2: int(y_train[:, 2].sum() * 1.5)}  # Tambahkan 50% sampel ke kelas 2

        # Initialize ADASYN with the dictionary and increased n_neighbors
        sampler = ADASYN(random_state=42, n_neighbors=3, sampling_strategy=sampling_strategy)
    else:
        raise ValueError("Invalid method. Choose 'SMOTE' or 'ADASYN'.")

    X_resampled, y_resampled = sampler.fit_resample(X_train, y_train)
    return X_resampled, y_resampled
```

Fig.4: *Oversampling candidate label.*



```
# Oversample Training Data using SMOTE or ADASYN
def oversample_data(X_train, y_train, method="SMOTE"):
    if method == "SMOTE":
        sampler = SMOTE(random_state=42, k_neighbors=3)
    elif method == "ADASYN":
        sampler = ADASYN(random_state=42, n_neighbors=3)
    else:
        raise ValueError("Invalid method. Choose 'SMOTE' or 'ADASYN'.")

    X_resampled, y_resampled = sampler.fit_resample(X_train, y_train)
    return X_resampled, y_resampled
```

Fig.5: *Oversampling sentiment label.*

and the oversampling method, this function receives input feature data. The final result of this function is new training data (X_resampled, y_resampled), which is already balanced.

2.7 Naïve Bayes

Naive Bayes (NB) is a probabilistic classifier based on Bayes' theorem with the assumption of conditional independence between each pair of features given the class variable. This assumption allows parameter factorization [25]. Despite this simplification, which is often violated in practice, NB often obtains competitive accuracy on various classification tasks. The independence assumption gives NB several advantages, such as efficient parameter learning [26], the absence of structure learning, and the ability to function effectively with relatively small amounts of data, since it only estimates bivariate statistics. These advantages contribute to the widespread use of NB. Equation 2 shows that μ and σ are estimates of the mean and standard deviation using the maximum likelihood principle [27].

$$P(v|y) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot \frac{(v-u)^2}{2\sigma^2} \tag{2}$$

2.8 Support Vector Machine

Support Vector Machine (SVM) is an algorithm that uses optimization theory and hypothetical space in a series of pattern recognition fields and uses kernels to map input space [28]. SVM is often implemented for various problems and purposes, such as pattern identification, bioinformatics, and text classification by describing the hyperplane as an input feature consisting of two classes, then re-optimized into more than two classes [29]. The unlimited function in hyperplane search in the Support Vector Machine method is an advantage, where processing will always be possible regardless of the data used. Equation 3 shows the algorithm in the hyperplane process.

$$f(x) = \sum\nolimits_{i=1}^{m} sign(\alpha_i y_i K(x\ x_i) + b) \tag{3}$$

2.9 K-Nearest Neighbors

Classification of datasets using the K-Nearest Neighbors (KNN) method. The main purpose of this algorithm is to divide objects based on attributes and training samples. The KNN algorithm uses proximity classification at points as an estimated value of the new query instance. This study uses cross-validation model validation for assessing KNN performance by dividing the K value into all parts of the dataset. The K part is used as test data in assessing KNN performance with a predetermined K range[30]. The formula for calculating the closeness between two cases can be seen in Equation 4.

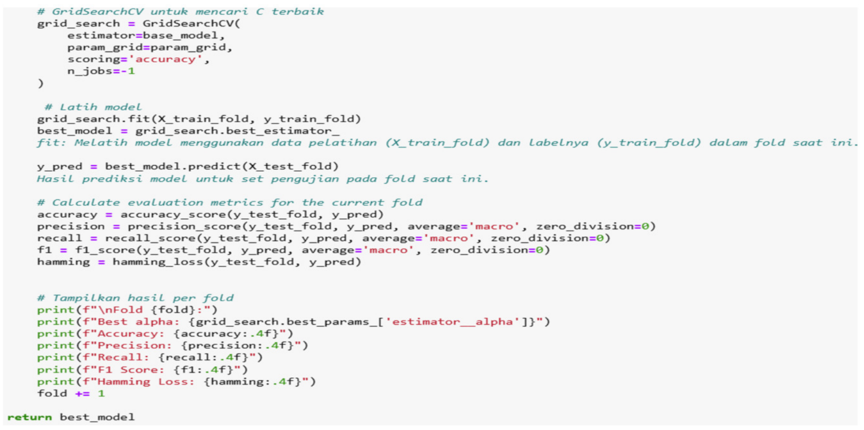
$$\frac{\sum_{i=1}^{n} \int (Ti, Si) * wi}{wi} \tag{4}$$

2.10 Cross Validation

Evaluation of the assessment details is carried out using the k-fold cross-validation method with a value of k = 10. This evaluation method is widely used for text classification [31]. Cross-validation (K-Fold) is a method used to evaluate predictions that are divided into training samples and test samples. Most of the data partitions are divided to train the model, and a small part is used for testing. After that, it will be repeated for a certain time so that the errors that occur each time can be identified [32]. The testing process in Figure 6 used is 10-fold cross-validation, which is iterated ten times in each test scenario with a combination of GridSearchCV.

2.11 Confusion Matrix

The confusion matrix is generally used to measure the performance of an algorithm [33].



```
    # GridSearchCV untuk mencari C terbaik
    grid_search = GridSearchCV(
        estimator=base_model,
        param_grid=param_grid,
        scoring='accuracy',
        n_jobs=-1
    )

     # Latih model
    grid_search.fit(X_train_fold, y_train_fold)
    best_model = grid_search.best_estimator_
    fit: Melatih model menggunakan data pelatihan (X_train_fold) dan labelnya (y_train_fold) dalam fold saat ini.

    y_pred = best_model.predict(X_test_fold)
    Hasil prediksi model untuk set pengujian pada fold saat ini.

    # Calculate evaluation metrics for the current fold
    accuracy = accuracy_score(y_test_fold, y_pred)
    precision = precision_score(y_test_fold, y_pred, average='macro', zero_division=0)
    recall = recall_score(y_test_fold, y_pred, average='macro', zero_division=0)
    f1 = f1_score(y_test_fold, y_pred, average='macro', zero_division=0)
    hamming = hamming_loss(y_test_fold, y_pred)


    # Tampilkan hasil per fold
    print(f"\nFold {fold}:")
    print(f"Best alpha: {grid_search.best_params_['estimator__alpha']}")
    print(f"Accuracy: {accuracy:.4f}")
    print(f"Precision: {precision:.4f}")
    print(f"Recall: {recall:.4f}")
    print(f"F1 Score: {f1:.4f}")
    print(f"Hamming Loss: {hamming:.4f}")
    fold += 1

return best_model
```

Fig.6: *K-fold 10 cross validation.*

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{5}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{6}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{7}$$

$$\text{F1-Score} = \frac{2 \times Recall \times Precision}{Recall + Precision} \tag{8}$$

The classified data is evaluated to obtain a Confusion matrix table consisting of variables such as True Positive (TN), False Positive (FP), True Negative (TN), and False Negative (FN) to calculate accuracy in Equation (5), precision in Equation (6), recall in Equation (7), and F1-score in Equation (8) [34].

3. RESULTS AND DISCUSSION

Multi-label classification is used for more complex opinion-breaking processes. Therefore, this study uses two categories for the Presidential and Governor Candidate datasets, namely sentiment labels and candidate labels. The results of this study show the

Table 3: *Accuracy and Precision of 2024 Presidential Candidates.*

2024 Presidential Candidate Labels	Accuracy	Precision (Anies Baswedan)	Precision (Prabowo Subianto)	Precision (Ganjar Pranowo)
Naive Bayes	0.891	0.92	0.94	0.95
Naive Bayes + SMOTE	0.891	0.92	0.94	0.95
Naive Bayes + ADASYN	0.893	0.92	0.94	0.94
Naive Bayes + SMOTE (GridSearchCV)	0.898	0.92	0.94	0.94
Naive Bayes + ADASYN (GridSearchCV)	0.902	0.92	0.94	0.94
SVM	0.942	0.97	0.97	0.97
SVM +SMOTE	0.937	0.96	0.98	0.98
SVM + ADASYN	0.939	0.97	0.95	0.96
SVM +SMOTE (GridSearchCV)	0.911	0.95	0.95	0.96
SVM + ADASYN (GridSearchCV)	0.945	0.97	0.96	0.98
KNN	0.729	0.92	0.82	0.94
KNN+SMOTE	0.738	0.96	0.80	0.93
KNN+ADASYN	0.902	0.97	0.92	0.96
KNN+SMOTE (GridSearchCV)	0.738	0.96	0.80	0.93
KNN+ADASYN (GridSearchCV)	0.902	0.97	0.92	0.96

Table 4: *Recall and F1-Score of 2024 Presidential Candidates.*

2024 Presidential Candidate Labels	Recall (Anies Baswedan)	Recall (Prabowo Subianto)	Recall (Ganjar Pranowo)	F1-Score (Anies Baswedan)	F1-Score (Prabowo Subianto)	F1-Score (Ganjar Pranowo)
Naive Bayes	0.94	0.86	0.91	0.93	0.90	0.93
Naive Bayes + SMOTE	0.94	0.86	0.91	0.93	0.90	0.93
Naive Bayes + ADASYN	0.95	0.86	0.92	0.93	0.90	0.93
Naive Bayes + SMOTE (GridSearchCV)	0.95	0.88	0.93	0.93	0.91	0.93
Naive Bayes + ADASYN (GridSearchCV)	0.95	0.88	0.93	0.94	0.91	0.94
SVM	0.96	0.94	0.95	0.97	0.95	0.96
SVM +SMOTE	0.97	0.92	0.93	0.97	0.95	0.96
SVM + ADASYN	0.97	0.95	0.95	0.97	0.95	0.96
SVM +SMOTE (GridSearchCV)	0.96	0.90	0.92	0.95	0.93	0.94
SVM + ADASYN (GridSearchCV)	0.97	0.95	0.95	0.97	0.95	0.96
KNN	0.68	0.83	0.68	0.78	0.82	0.79
KNN+SMOTE	0.60	0.88	0.74	0.74	0.84	0.83
KNN+ADASYN	0.89	0.93	0.89	0.93	0.92	0.92
KNN+SMOTE (GridSearchCV)	0.60	0.88	0.74	0.74	0.84	0.83
KNN+ADASYN (GridSearchCV)	0.89	0.93	0.89	0.93	0.92	0.92

k-fold evaluation process and comparison between the three NB, SVM, and KNN algorithms, with the results of two categories of sentiment and candidate labels using GridSearchCV or without GridSearchCV. Then the final result of the highest accuracy from the comparison of the three algorithms using GridSearchCV.

3.1 Method Performance

This study aims to measure the effectiveness of three main classification algorithms, Naive Bayes, SVM, and KNN, in predicting candidate choices and public sentiment towards presidential and gubernatorial candidates for DKI Jakarta in the 2024 Election. The presidential election dataset was obtained from Azno *et al.* [5] where the dataset was examined using several comparative methods, such as SVM - Linear, SVM - Polynomial, SVM - RBF, SVM - Sigmoid and Naïve Bayes. Experiments were conducted with data split ratios of 70:30, 80:20, and 90:10. The best results were obtained with the SVM - RBF method with accuracy values of 86, 85, and 87, for each split ratio. The gubernatorial election dataset, meanwhile, was the result of crawling Twitter for this study.

In this research, testing was carried out using the SMOTE and ADASYN data weighting techniques, and a comparison was made between model performance before and after hyperparameter tuning using GridSearchCV in Tables 4-7 for the presidential candidate dataset and Tables 8-11 for the gubernatorial candidate dataset. In Tables 3 and 4 for presidential candidate classification, the Naive Bayes algorithm without oversampling showed good initial results with an accuracy of 0.891, high precision Anies 0.92, Prabowo 0.94, Ganjar 0.95 and a stable F1-score average 0.92. However, recall for Prabowo was still low 0.86, indicating an imbalance in class detection. After fine-tuning with GridSearchCV, specifically the NB + ADASYN combination, accuracy increased to 0.902. This increase was also followed by improvements in recall for Prabowo 0.88 and Ganjar 0.93, as

Table 5: *Accuracy and Precision of 2024 Presidential Sentiment.*

2024 Presidential Sentiment Labels	Accuracy	Precision (Positive)	Precision (Negative)
Naive Bayes	0.866	0.87	0.84
Naive Bayes + SMOTE	0.901	0.95	0.79
Naive Bayes + ADASYN	0.880	0.92	0.77
Naive Bayes + SMOTE (GridSearchCV)	0.918	0.96	0.83
Naive Bayes + ADASYN (GridSearchCV)	0.897	0.93	0.81
SVM	0.973	0.98	0.95
SVM + SMOTE	0.968	0.97	0.96
SVM + ADASYN	0.968	0.98	0.93
SVM +SMOTE (GridSearchCV)	0.986	0.99	0.98
SVM + ADASYN (GridSearchCV)	0.987	1.00	0.97
KNN	0.641	0.96	0.43
KNN+SMOTE	0.622	0.99	0.42
KNN+ADASYN	0.608	1.00	0.41
KNN+SMOTE (GridSearchCV)	0.622	0.99	0.42
KNN+ADASYN (GridSearchCV)	0.608	1.00	0.41

Table 6: *Recall and F1-Score of 2024 Presidential Sentiment.*

2024 Presidential Sentiment Labels	Recall (Positive)	Recall (Negative)	F1-Score (Positive)	F1-Score (Negative)
Naive Bayes	0.96	0.62	0.91	0.72
Naive Bayes + SMOTE	0.91	0.88	0.93	0.83
Naive Bayes + ADASYN	0.91	0.80	0.92	0.78
Naive Bayes + SMOTE (GridSearchCV)	0.93	0.89	0.94	0.86
Naive Bayes + ADASYN (GridSearchCV)	0.93	0.81	0.93	0.81
SVM	0.98	0.95	0.98	0.95
SVM + SMOTE	0.99	0.92	0.98	0.94
SVM + ADASYN	0.97	0.96	0.98	0.94
SVM +SMOTE (GridSearchCV)	0.99	0.97	0.99	0.98
SVM + ADASYN (GridSearchCV)	0.99	0.99	0.99	0.98
KNN	0.53	0.94	0.68	0.59
KNN+SMOTE	0.49	0.98	0.65	0.59
KNN+ADASYN	0.46	0.99	0.63	0.58
KNN+SMOTE (GridSearchCV)	0.49	0.98	0.65	0.59
KNN+ADASYN (GridSearchCV)	0.46	0.99	0.63	0.58

well as an increase in the overall F1-score. This indicates that the NB + ADASYN + GridSearchCV combination provided the best results, making the model more balanced in recognizing all candidates. On the other hand, the SVM algorithm without oversampling already performed well, with an accuracy of 0.942 and precision and recall values for all three candidates above 0.94, indicating that the default SVM parameters were highly optimal. After tuning, the best combination of SVM + ADASYN + GridSearchCV produced an accuracy of 0.945, a slight improvement, indicating that SVM is quite reliable even without tuning. In contrast, the KNN algorithm without oversampling performed poorly, with an accuracy of only 0.729, recalls for Anies and Ganjar of 0.68 each, and an F1-score below 0.80. This indicates that KNN relies heavily on data balancing techniques. When using SMOTE, accuracy increased slightly to 0.738, but remained low. The KNN + ADASYN combination actually provided a significant jump, reaching an accuracy of 0.902, and this result remained stable after GridSearchCV. These findings confirm that data distribution imbalance is crucial to the effectiveness of KNN.

In the presidential candidate sentiment classification shown in Tables 5 and 6, Naive Bayes without oversampling demonstrated an accuracy of 0.866, with a negative recall of only 0.62 and a negative F1-score of 0.72, indicating weakness in recognizing negative sentiment. With GridSearchCV + SMOTE, the accuracy increased to 0.918, and the negative recall improved to 0.89, making the model more balanced. Meanwhile, SVM without oversampling already produced excellent results with an accuracy of 0.973, and

Table 7: *Accuracy and precision of 2024 Gubernatorial Candidates.*

Jakarta Gubernatorial Candidate Labels 2024	Accuracy	Precision (Pramono Anung)	Precision (Ridwan Kamil)	Precision (Dharma Pongrekun)
Naive Bayes	0.931	0.93	0.92	0.99
Naive Bayes + SMOTE	0.931	0.93	0.92	0.99
Naive Bayes + ADASYN	0.939	0.97	0.90	0.99
Naive Bayes + SMOTE (GridSearchCV)	0.938	0.94	0.93	0.99
Naive Bayes + ADASYN (GridSearchCV)	0.945	0.98	0.91	0.99
SVM	0.974	0.98	0.96	1.00
SVM + SMOTE	0.970	0.97	0.95	1.00
SVM + ADASYN	0.969	0.98	0.94	0.99
SVM +SMOTE (GridSearchCV)	0.970	0.97	0.96	1.00
SVM + ADASYN (GridSearchCV)	0.969	0.98	0.94	0.99
KNN	0.909	0.96	0.82	0.99
KNN+SMOTE	0.906	0.96	0.81	0.99
KNN+ADASYN	0.971	0.97	0.95	0.99
KNN+SMOTE (GridSearchCV)	0.906	0.96	0.81	0.99
KNN+ADASYN (GridSearchCV)	0.971	0.97	0.95	0.99

Table 8: *Recall and F1-Score of 2024 Gubernatorial Candidates.*

Jakarta Gubernatorial Candidate Labels 2024	Recall (Pramono Anung)	Recall (Ridwan Kamil)	Recall (Dharma Pongrekun)	F1-Score (Pramono Anung)	F1-Score (Ridwan Kamil)	F1-Score (Dharma Pongrekun)
Naive Bayes	0.96	0.97	0.99	0.94	0.94	0.99
Naive Bayes + SMOTE	0.96	0.97	0.99	0.94	0.94	0.99
Naive Bayes + ADASYN	0.91	0.97	0.99	0.94	0.94	0.99
Naive Bayes + SMOTE (GridSearchCV)	0.96	0.98	1.00	0.95	0.95	0.99
Naive Bayes + ADASYN (GridSearchCV)	0.92	0.98	0.99	0.95	0.94	0.99
SVM	0.96	0.97	1.00	0.97	0.97	1.00
SVM + SMOTE	0.96	0.98	1.00	0.96	0.97	1.00
SVM + ADASYN	0.94	0.98	1.00	0.96	0.96	1.00
SVM + SMOTE (GridSearchCV)	0.96	0.98	1.00	0.97	0.97	1.00
SVM + ADASYN (GridSearchCV) heightKNN	0.94 0.80	0.98 0.96	1.00 0.96	0.96 0.88	0.96 0.88	1.00 0.98
KNN+SMOTE	0.80	0.96	0.96	0.87	0.88	0.98
KNN+ADASYN	0.95	0.97	1.00	0.96	0.96	0.99
KNN+SMOTE (GridSearchCV)	0.80	0.96	0.96	0.87	0.88	0.98
KNN+ADASYN (GridSearchCV)	0.95	0.97	1.00	0.96	0.96	0.99

all precision, recall, and F1-score metrics for positive and negative labels approached or reached 0.98-0.99. Parameter tuning only improved the accuracy to 0.987, indicating that the model was optimal from the start. In contrast, KNN without oversampling performed very poorly, with an accuracy of 0.641, a positive recall of 0.53, and a negative F1-score of only 0.59. Balancing techniques like ADASYN failed to significantly improve performance its accuracy even dropped to 0.608, and GridSearchCV had no significant impact, especially in negative sentiment classification. This demonstrates the limitations of KNN in handling imbalanced data distributions.

In the classification of Jakarta gubernatorial candidates Tables 7 and 8, Naive Bayes without oversampling demonstrated excellent initial performance with an accuracy of 0.931, precision and recall for all candidates above 0.92, and an average F1-score above 0.94. Tuning with ADASYN + GridSearchCV increased accuracy to 0.945, indicating a small but significant improvement, particularly in recall for candidates Ridwan Kamil and Pramono Anung. SVM without oversampling again performed superiorly with an accuracy of 0.974, recall for Dharma Pongrekun reaching 1.00, and F1-scores nearing perfection for all candidates. After fine-tuning, accuracy remained high without significant changes, indicating that SVM is highly effective and consistent for multi-label classification with relatively balanced data distribution. In contrast, KNN without oversampling provided an accuracy of 0.909, lower than SVM and Naive Bayes, and Pramono's recall was only 0.80. Using ADASYN increased accuracy to 0.971, indicating that data balancing plays a significant role in KNN effectiveness. However, when using SMOTE, KNN's accuracy remained low 0.906, indicating the model's sensitivity to the oversampling technique used.

In the sentiment classification for gubernatorial candidates, Tables 9 and 10 show an accuracy of 0.914 without oversampling, with balanced recall and F1-score for both labels 0.91-0.92, making it a fairly stable model from the start. After GridSearchCV +

Table 9: *Accuracy and Precision of 2024 Gubernatorial Sentiment.*

Jakarta Gubernatorial Candidate Labels 2024	Accuracy	Precision (Positive)	Precision (Negative)
Naive Bayes	0.914	0.93	0.90
Naive Bayes + SMOTE	0.891	0.94	0.84
Naive Bayes + ADASYN	0.885	0.95	0.82
Naive Bayes + SMOTE (GridSearchCV)	0.905	0.95	0.86
Naive Bayes + ADASYN (GridSearchCV)	0.900	0.96	0.84
SVM	0.985	0.98	0.99
SVM + SMOTE	0.982	0.98	0.99
SVM + ADASYN	0.983	0.99	0.98
SVM +SMOTE (GridSearchCV)	0.992	0.99	1.00
SVM + ADASYN (GridSearchCV)	0.992	0.99	0.99
KNN	0.943	0.98	0.91
KNN+SMOTE	0.930	0.98	0.88
KNN+ADASYN	0.918	0.99	0.85
KNN+SMOTE (GridSearchCV)	0.930	0.98	0.88
KNN+ADASYN (GridSearchCV)	0.918	0.99	0.85

Table 10: *Recall and F1-Score of 2024 Gubernatorial Sentiment.*

Jakarta Gubernatorial Candidate Labels 2024	Recall (Positive)	Recall (Negative)	F1- Score (Positive)	F1-Score (Negative)
Naive Bayes	0.92	0.91	0.92	0.91
Naive Bayes + SMOTE	0.86	0.94	0.90	0.89
Naive Bayes + ADASYN	0.83	0.95	0.89	0.88
Naive Bayes + SMOTE (GridSearchCV)	0.87	0.95	0.91	0.90
Naive Bayes + ADASYN (GridSearchCV)	0.85	0.96	0.90	0.90
SVM	0.99	0.98	0.99	0.98
SVM + SMOTE	0.99	0.98	0.98	0.98
SVM + ADASYN	0.98	0.98	0.98	0.98
SVM +SMOTE (GridSearchCV)	1.00	0.99	0.99	0.99
SVM + ADASYN (GridSearchCV)	1.00	0.99	0.99	0.99
KNN	0.92	0.97	0.95	0.94
KNN+SMOTE	0.89	0.98	0.93	0.93
KNN+ADASYN	0.86	0.99	0.92	0.92
KNN+SMOTE (GridSearchCV)	0.89	0.98	0.93	0.93
KNN+ADASYN (GridSearchCV)	0.86	0.99	0.92	0.92

ADASYN, the accuracy decreased slightly to 0.900-0.905, but the recall for negative sentiment increased from 0.91 to 0.96, indicating the model became more sensitive to the minority class. SVM without oversampling performed best with an accuracy of 0.985, with very high recall and F1-score for both positive and negative sentiment. After tuning, the accuracy reached 0.992, and the recall for positive sentiment reached 1.00, indicating that parameter tuning optimally improved the model's sensitivity and specificity. Meanwhile, KNN without oversampling showed quite high performance with an accuracy of 0.943, a positive recall of 0.92, and a negative recall of 0.97, making it quite competitive. However, after using ADASYN, the accuracy actually decreased to 0.918, and the results remained stable after tuning. This shows that while KNN can achieve good results on balanced data, it still has limitations in handling complex sentiment classification, especially if the data balancing is not done properly.

From all the analysis results, it can be concluded that SVM is the most superior algorithm in all types of classification, both candidates and sentiment. This model provides high and stable accuracy and F1-score, even before parameter tuning. Later, Naive Bayes is a powerful and efficient alternative, with performance that consistently increases after the application of GridSearchCV, especially in handling unbalanced data distributions. Furthermore, KNN is only suitable for local classification, such as in the Jakarta gubernatorial election, and is only effective when combined with balancing techniques such as

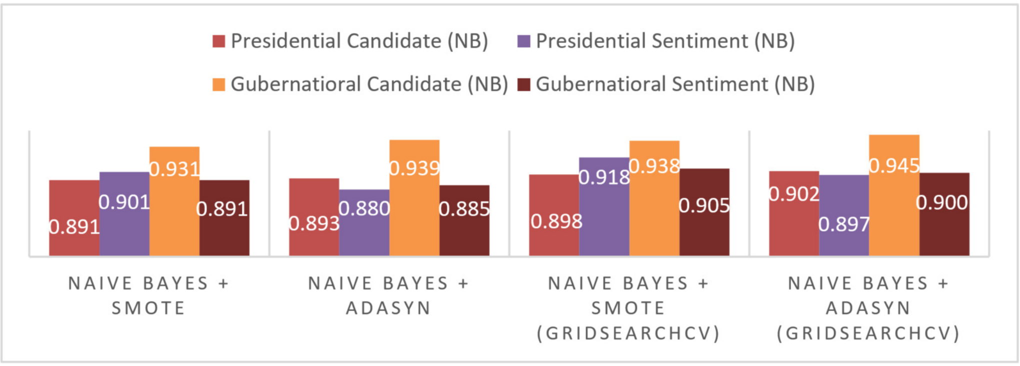


Fig.7: *Naive Bayes Comparison of Presidential and Gubernatorial Candidates.*

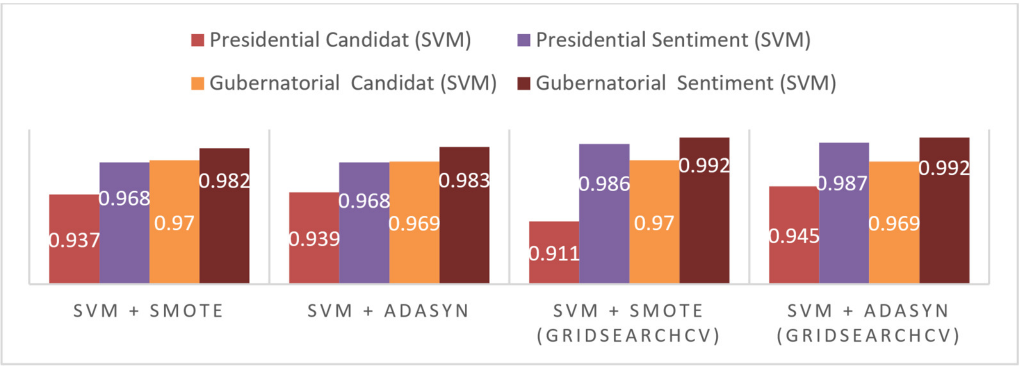


Fig.8: *SVM Comparison of Presidential and Gubernatorial Candidates.*

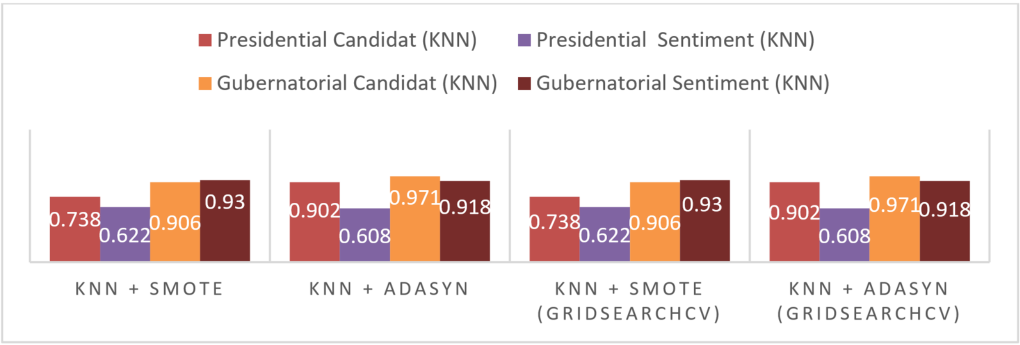


Fig.9: *KNN Comparison of Presidential and Gubernatorial Candidates.*

ADASYN. For data classification with high complexity, KNN is less recommended because its performance is unstable and tends to be lower than other algorithms.

3.2 Comparison NB, SVM and KNN

The following is an explanation of the results of the comparative accuracy of the Naive Bayes, SVM, and KNN methods based on Figures 7 to 8. This analysis includes a comparison of performance before and after using GridSearchCV on sentiment and candidate labels, both for the presidential and gubernatorial elections of DKI Jakarta.

In Figure 7, it can be seen that the Naive Bayes model experienced a significant increase in accuracy after parameter tuning using GridSearchCV. For candidate label classification, the initial accuracy was in the range of 0.891-0.894, and increased to 0.901, especially when using the ADASYN method as an oversampling technique. A similar increase also occurred in sentiment classification, where the accuracy increased from 0.880-0.901 to 0.898-0.918 after tuning. This shows that Naive Bayes is quite sensitive to parameters and gets a significant performance increase through the tuning process. In addition, the use of ADASYN is proven to provide more consistent and balanced results compared to SMOTE, both for candidate and sentiment data.

Meanwhile, Figure 8 shows that SVM is the algorithm with the best and most stable performance among the three models. In candidate label classification, SVM shows high accuracy from the beginning, around 0.930-0.933, and only slightly increases after tuning. However, a more significant increase is seen in sentiment classification, where the accuracy increases from 0.968 to 0.992 after parameter tuning with a combination of SVM + SMOTE + GridSearchCV. This indicates that even though SVM is very optimal with default parameters, the GridSearchCV process is still able to improve performance, especially on more complex data such as sentiment analysis.

In contrast to the two algorithms, Figure 9 shows that KNN is the most unstable algorithm and is highly dependent on the oversampling method used. The combination of KNN + SMOTE produces low accuracy, ranging from 0.622 to 0.741, both before and after GridSearchCV. This indicates that SMOTE is less effective when combined with KNN on this data. In contrast, the combination of KNN + ADASYN produces much better and more stable results, with accuracies reaching 0.903 to 0.971 for candidate classification, and 0.917 to 0.930 for sentiment classification. However, GridSearchCV does not provide a significant improvement to KNN performance, indicating that KNN performance is more determined by the data balancing strategy than by parameter tuning.

Overall, the visualizations in Figures 7 to 9 confirm previous findings that SVM is the most superior algorithm, providing the best and most consistent performance in all types of classification, both before and after parameter tuning. Later, Naive Bayes improves significantly after tuning and works particularly well when combined with ADASYN. Furthermore, KNN shows the most unstable performance and is highly dependent on the oversampling method. GridSearchCV does not contribute much to improving the performance of KNN, so the effectiveness of this model is more determined by the selection of the right data balancing technique.

4. CONCLUSIONS

Overall, SVM shows the best and most stable results across all classification and sentiment contexts, both before and after tuning. Naive Bayes provides fairly competitive results, especially after tuning and when used with oversampling methods such as ADASYN. In contrast, KNN relies heavily on data balancing techniques and shows unstable performance, especially when used with SMOTE. The application of GridSearchCV generally has a positive impact, especially for Naive Bayes and in some cases for SVM, while the effect is relatively small or insignificant for KNN.

ACKNOWLEDGEMENT

This research was supported by Universitas Ahmad Dahlan.

AUTHOR CONTRIBUTIONS

Conceptualization, H.Y., R.A. and A.Y.; Methodology, H.Y., R.A. and A.Y.; Software, R.A.; Validation, H.Y. and R.A.; Formal analysis, H.Y. and R.A.; Investigation, H.Y., R.A. and A.Y.; Resources, R.A. and A.Y.; Data Curation, H.Y., R.A. and A.Y.; Writing - Original Draft, H.Y. and R.A.; Writing - Review & Editing, H.Y., R.A. and A.Y.; Visualization, R.A.; Supervision, H.Y. and A.Y.; Project administration, H.Y.; Funding acquisition, H.Y.

References

[1] K. Munawaroh and Alamsyah, "Performance Comparison of SVM , Naïve Bayes , and KNN Algorithms for Analysis of Public Opinion Sentiment Against COVID-19 Vaccination on Twitter," *Journal of Advances in Information Systems and Technology*, vol. 4, no. October, pp. 113–125, 2022.

[2] M. Farhan, Manik, H. R. Jannah and L. H. Suadaa, "Comparison of Naive Bayes, K-Nearest Neighbor, and Support Vector Machine Classification Methods in Semi-Supervised Learning for Sentiment Analysis of Kereta Cepat Jakarta Bandung (KCJB)," in *Proceedings of 2023 International Conference on Data Science and Of-*

ficial Statistics (ICDSOS), vol. 2023, no. 1, pp. 109–120, 2023.

[3] N. Mardiah, L. Marlina, K. Khairul, Z. Sitorus and M. Iqbal, "Analysis Of Indonesian People's Sentiment Towards 2024 Presidential Candidates On Social Media Using Naïve Bayes Classifier and Support Vector Machine," *Building of Informatics, Technology and Science (BITS)*, vol. 6, no. 2, pp. 950–960, 2024.

[4] A. A. Firdaus, A. Yudhana and R. Imam, "Analisis Sentimen Pada Proyeksi Pemilihan Presiden 2024 Menggunakan Metode Support Vector Machine," *DECODE: Jurnal Pendidikan Teknologi Informas*, vol. 3, no. 2, pp. 236–245, 2024.

[5] A. A. Firdaus, A. Yudhana and I. Riadi, "Prediction of Presidential Election Results using Sentiment Analysis with Pre and Post Candidate Registration Data," *Jurnal Ilmu Komputer dan Informatika*, vol. 10, no. 1, pp. 36–46, 2024.

[6] A. A. Firdaus, A. Yudhana, I. Riadi and Mahsun, "Indonesian presidential election sentiment: Dataset of response public before 2024," *Data in Brief*, vol. 52, p. 109993, Feb. 2024.

[7] W. LI, S. Zhu, Z. Li and H. Wang, "Kernel-Based Enhanced Oversampling Method for Imbalanced Classification," *arXiv preprint* arXiv:2504.09147v1, 2025.

[8] S. Akter, I. Ishika, P. R. Das, M. Julker Nyne and D. M. Farid, "Boosting Oversampling Methods for Imbalanced Data Classification," *2023 26th International Conference on Computer and Information Technology (ICCIT)*, Cox's Bazar, Bangladesh, pp. 1-6, 2023.

[9] N. K. Rajput, V. K. Rathi, B. A. Grover and R. Bansal, "Word Frequency and Sentiment Analysis of Twitter," *arXiv preprint* arXiv:2004.03925v2, 2024.

[10] J. Hemmatian, R. Hajizadeh and F. Nazari, "Addressing imbalanced data classification with Cluster-Based Reduced Noise SMOTE," *PLoS ONE*, vol. 20, no. 2, p. e0317396, 2025.

[11] S. A. Chaurasia and S. S. Sherekar, "Sentiment Analysis of Twitter Data by Natural Language Processing and Machine Learning Sentiment Analysis of Twitter data by Natural Language Processing and Machine Learning," *Proceedings of International Conference on Advanced Communications and Machine Intelligence*, pp. 1–15, 2023.

[12] K. S. Eljil, F. Naït-Abdesselam, E. Hamouda and M. Hamdi, "Enhancing Sentiment Analysis on Social Media with Novel Preprocessing Techniques," *Journal of Advances in Information Technology*, vol. 14, no. 6, pp. 1206–1213, 2023.

[13] M. D. Samad, N. Khounviengxay and M. A. Witherow, "Effect of Text Processing Steps on Twitter Sentiment Classification using Word Embedding," *arXiv preprint* arXiv:2007.13027v1, 2020.

[14] S. S. Berutu, H. Budiati, Jatmika and F. Gulo, "Data preprocessing approach for machine learning-based sentiment classification," *INFOTEL (Informatics, Telecommunication, and Electronics)*, vol. 15, no. 4, pp. 317–325, 2023.

[15] M. A. Palomino and F. Aider, "Evaluating the Effectiveness of Text Pre-Processing in Sentiment Analysis," *Applied Sciences*, vol. 12, no. 17, p. app12178765, 2022.

[16] H.-T. Duong and T.-A. Nguyen-Thi, "A review:preprocessing techniques and data augmentation for sentiment analysis," *Computational Social Networks*, vol. 8, no. 1, pp. 1–16, 2021.

[17] F. Carvalho and G. P. Guedes, "TF-IDFC-RF : A Novel Supervised Term Weighting Scheme for Sentiment Analysis," *arXiv preprint* arXiv:2003.07193v2, 2020.

[18] J. Frej, P. Mulhem, S. Didier and J.-P. Chevallet, "Learning Term Discrimination," *Association for Computing Machinery*, pp. 18–21, 2020.

[19] Y. Cathy, D. Paul, W. Jörg and T. Katerina, "A systematic review of aspect-based sentiment analysis:domains, methods, and trends, *Artificial Intelligence Review*, Springer Netherlands, vol. 57, no. 296, 2024.

[20] E. Xu, J. Zhu, L. Zhang, Y. Wang and W. Lin, "Research on Aspect-Level Sentiment Analysis Based on Adversarial Training and Dependency Parsing," *Electronics*, vol. 13, no. 10, p. 13101993, 2024.

[21] A. R. Salehi and M. Khedmati, "A cluster-based SMOTE both-sampling (CSBBoost) ensemble algorithm for classifying imbalanced data," *Scientific Reports*, vol.14, no. 5152, 2024.

[22] Y. Zhang, L. Deng and B. Wei, "Imbalanced Data Classification Based on Improved Random-SMOTE and Feature Standard Deviation," *Mathematics*, vol. 12, no. 11, p. math12111709, 2024.

[23] A. Taskeen, S. U. R. Khan and A. Mashkoor,"An adaptive synthetic sampling and batch generation-oriented hybrid approach for addressing class imbalance problem in software defect prediction," *Soft Computing* , vol. 28, no. 23–24, pp. 13595–13614, 2024.

[24] M. K. Zuhanda, L. Permata, Hartono, E. Ongko and Desniarti, "Impact of Adaptive Synthetic on Naïve Bayes Accuracy in Imbalanced Anemia Detection Datasets," *Rekayasa Sistem dan Teknologi Informasi*, vol. 5, no. 158, pp. 4–12, 2025.

[25] P. Torrijos, J. C. Alfaro, J. A. Gámez and J. M. Puerta, "Federated Learning with Discriminative Naive Bayes Classifier," *arXiv preprint* arXiv:2502.01532v1, 2025.

[26] O. Peretz, M. Koren and O. Koren, "Naive Bayes

classifier-An ensemble procedure for recall and precision enrichment," *Engineering Applications of Artificial Intelligence*, vol. 136, no. PB, p. 108972, 2024.

[27] H. Chen, S. Hu, R. Hua, and X. Zhao, "Improved naive Bayes classification algorithm for traffic risk management," *EURASIP Journal on Advances in Signal Processing*, vol. 2021, no. 30, 2021.

[28] Z. Jun, "The Development and Application of Support Vector Machine," *Journal of Physics: Conference Series*, vol. 1748, pp. 1–7, 2021.

[29] I. Hossain, "Support Vector Machine," *ResearchGate*, pp. 1–7, 2022.

[30] E. Bayraktar, I. Ekren and X. Zhang, "Prediction against a limited adversary," *Journal of Machine Learning Research*, vol. 22, pp. 1–33, 2021.

[31] J. Qiu, "An Analysis of Model Evaluation with Cross-Validation:Techniques , Applications , and Recent Advances," *Proc. Financ. Age Environ. Risks Sustain.*, pp. 69–72, 2024.

[32] V. W. Lumumba and D. Kiprotich, "Comparative Analysis of Cross-Validation Techniques: LOOCV , K-folds Cross-Validation , and Repeated K-folds Cross-Validation in Machine Learning Models,"*American Journal of Theoretical and Applied Statistics*, vol. 13, no. 5, pp. 127–137, 2024.

[33] S. Sathyanarayanan and B. R. Tantri, "Confusion Matrix-Based Performance Evaluation Metrics," *African Journal of Biomedical Research*, vol. 27, no. 4, pp. 1–9, 2024.

[34] A. F. Alshammari, "Implementation of Model Evaluation using Confusion Matrix in Python," *International Journal of Computer Applications*, vol. 186, no. 50, pp. 42–48, 2024.

Herman Yuliansyah received the S.T. degree from Faculty of Engineering, Universitas Muhammadiyah Yogyakarta (UMY) Indonesia in 2007, M.Eng degree in Information Technology from Universitas Gajah Mada (UGM) Indonesia in 2011. PhD degree in Artificial Intelligence from Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM) since 2019. He has worked in Department of Informatics, Universitas Ahmad Dahlan (UAD) Indonesia as lecturer since 2011. His research primarily focuses on Link Prediction, Social Computing, Text Mining and Social Network Analytics.

Ricy Ardiansyah is a student at the Department of Informatics, undergraduate program, since 2024. He is currently pursuing a Magister Informatika. degrees from the Faculty of Industrial Technology, Universitas Ahmad Dahlan (UAD), Indonesia. Ricy Ardiansyah is currently interested in the field of artificial intelligence research, especially text mining.

Anton Yudhana received his Bachelor's degree in Electrical Engineering from the Sepuluh Nopember Institute of Technology in 2001, his Master's degree in Electrical Engineering from Gadjah Mada University in 2005, and his Doctorate from the Universiti Teknologi Malaysia in 2011. He currently serves as an Associate Professor in the Department of Electrical Engineering at Ahmad Dahlan University, Indonesia, and as the Head of the Center for Electrical and Electronic Research and Development (CEERD). His current research interests include precision agriculture, biomedical engineering, and food engineering based on the Internet of Things