Vol. 4 No. (1) (June 2026) 1-18

Journal of Data Insights

http://journalnew.unimus.ac.id/index.php/jodi

Sentiment Analysis of FlyGaruda Review Using Support Vector Machine and Naive Bayes Algorithm

Gibran Masta Pangestu Baskoro[1], Galet Guntoro Setiaji[2], Ahmad Rifa'i[3]

[1,2]Department of Informatics Engineering, University Semarang, Indonesia

[3]Department of Information Systems, University Semarang, Indonesia

DOI: https://doi.org/10.26714/jodi.v4i1.1117

Article Info

Article history:
Received May 06, 2026
Revised May 21, 2026
Accepted June 01, 2026

Keywords:
FlyGaruda; sentiment analysis; Support Vector Machine; Multinomial Naive Bayes; TF-IDF.

Abstract

FlyGaruda is an official digital application owned by Garuda Indonesia that provides ticket booking and online check-in services for users. The sentiment analysis in this experiment was done by determining the efficiency of the Support Vector Machine and Multinomial Naive Bayes on reviews found on the Google Play Store. Methods employed for the purpose of the research include scraping, text processing, extraction of the TF-IDF feature, and evaluation through the Confusion Matrix. A total number of 4,891 reviews were obtained from the Google Play Store. After the preprocessing stage, 101 reviews were removed because their contents became empty following text cleaning procedures, resulting in a final dataset of 4,790 reviews used for classification. This reduction occurred because 101 rows were removed after becoming blank during the text cleaning process. The results showed that both models obtained an accuracy of 82.25%. However, the Support Vector Machine produces a weighted precision of 77.66% and an F1-Score of 78.91%, better at handling data imbalances. Meanwhile, Multinomial Naive Bayes excels in computing efficiency with a training time of 0.08 seconds compared to 90.60 seconds on the Support Vector Machine. In conclusion, although it is slower, the Support Vector Machine provides more consistent and accurate classification performance. This research contributes to the development of a machine learning-based opinion analysis system to improve the quality of aviation digital services in a sustainable manner. These findings can serve as a reference in the selection of the best algorithms between accuracy and computational speed in large text data and support data-driven decision-making in the modern air transportation industry in the current era of global sustainable digital transformation

✉ Correspondence Address:
E-mail: gibranmastapb13@gmail.com

e-ISSN: 2988 - 2109

INTRODUCTION

In the midst of the rapid development of digital transformation, the existence of software distribution platforms such as the Google Play Store has evolved into an interactive ecosystem that has a strategic role for various companies in understanding the level of customer satisfaction in a more in-depth and measurablemanner[1]. The platform not only serves as a place for application distribution, but also serves as a two-way communication space between users and developers. Every user-provided review is no longer just a brief meaningless opinion, but has turned into a valuable data set that represents the public's perception of the quality of service, user experience, and functional performance of an application [2].

In this context, user review data has a very high value because it is spontaneous, honest, and comes directly from the user experience [3]. Therefore, in the era of data-driven decision-making, the ability to process and analyze the review data is an important factor in strategic decision-making, especially for companies engaged in the technology-based and transportation services sector [4].

In an air transportation industry that demands high service standards, such as airlines, the ability to quickly respond to customer complaints is a crucial element in maintaining user reputation and loyalty [5]. In this case, thousands of user opinions spread in the form of unstructured text become a source of strategic insight that is of great value if it can be processed using an appropriate and systematic analytical approach [6].

As a form of digital service transformation implementation, Garuda Indonesia's national airline has presented an official digital platform called *FlyGaruda*. This application is designed to provide convenience for passengers to access various flight services independently and efficiently. Through this application, users can order tickets, select seats, do web check-ins, and access various other support services in real-time [7]. The presence of this application is an important step in improving operational efficiency while strengthening the customer experience in the digital era [8].

Nevertheless, while the *FlyGaruda* app offers a wide range of conveniences, user responses to the service show a wide variety of variations. The reviews provided not only contain positive appreciation, but also include technical criticism, system-related complaints, and input for feature improvements. As a result, the volume of user review data has become very large and continues to grow exponentially [9].The information filtering process becomes inefficient because it takes a long time, a lot of energy, and has the potential to cause subjective bias and human error. At massive data scales, the manual approach is no longer relevant to be used as the primary method in user sentiment analysis [10].

The above mentioned issues are better answered using a text mining and AI-based solution, more specifically a machine learning-based solution. Using the sentiment analysis method, computers can be programmed to read, understand, and automatically categorize thousands of reviews into sentiments that are positive, negative, and neutral.

The choice of algorithm plays a very critical role in the area of NLP. Some of the most common and compared algorithms for use in text classification include Support Vector Machine (SVM) and Multinomial Naive Bayes.

The advantages of both methods have been proven through various previous studies. Research by Maulana et al. [11] on the digital investment platform Pluang shows that the comparison between SVM and Naive Bayes results in a highly competitive performance in classifying sentiment polarity. Similar results were also found in the Java et al. study.[12] which analyzed sentiment on the social media app Threads, where both algorithms showed fairly balanced capabilities in various testing scenarios

In the public service and government sectors, a study by Suryono [13] on the National Police Super App and the research of Baihaqi et al. [14] The Deepseek application also shows that the comparative approach between SVM and Naive Bayes is capable of providing a clear picture of the

effectiveness of each model in text classification. Meanwhile, Anggara's research [15] on the administrative service system of Al Ihsan Hospital also strengthens the view that the comparison of these two algorithms is an important step in determining the best classification model in a text-based system. [16].

Previous studies on sentiment analysis of application reviews have generally focused only on overall accuracy performance without providing detailed class-wise evaluation, particularly for minority sentiment classes such as neutral sentiment. Several studies also primarily evaluated a single algorithm or did not clearly explain model configurations and preprocessing stages, making reproducibility difficult. In addition, the issue of imbalanced sentiment distribution in user review datasets is still rarely discussed in depth. Thus, this research is conducted to investigate the difference between Support Vector Machine and Multinomial Naïve Bayes with TF-IDF feature extraction with respect to class wise evaluation measures and computation time to achieve a broader insight regarding the performance of both algorithms on imbalanced reviews.[23].

In this research, SVM is going to be compared with Naïve Bayes. The data for Naïve Bayes was taken from the comments of the Flygaruda application. The TF-IDF method to perform feature extraction. After that, The confusion matrix to analyze all the results.[30].

METHOD

This procedure will involve several steps, starting from the extraction process of data from Google Play Store reviews flygaruda. This will be followed by the classification of the data according to the sentiments, positive, neutral, and negative. Following this is the preprocessing process, which includes case folding, tokenization, removal of stopword, and stemming. Subsequently, the process will continue with the extraction of data through the TF-IDF. The next step will see the division of data to 80:20, after which the use of SVM and Naive Bayes algorithm for training and testing mode will be carried out figure 1.

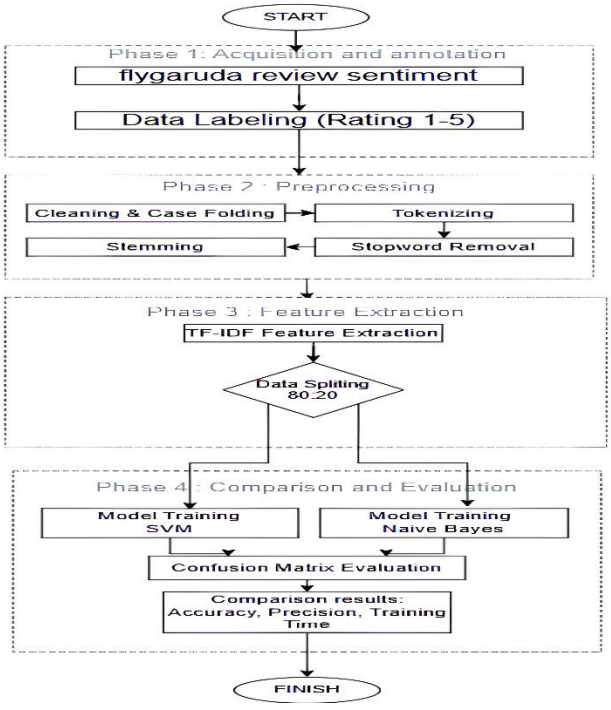

Figure1. Flygaruda Research Flow Diagram

the first is acquisition and annotation, then preprocessing, after that feature extraction and the final phase is comparison and evaluation.then using the Support Vector Machine (SVM) and Naive Bayes algorithms for training and testing modes. After that, just compare the two algorithms.

2.1 Scraping Data

The first phase of this research involves the process of collecting data from reviews provided by users of the FlyGaruda application found at Google Play Store. This is done through a process of web scraping which is an automated way of collecting data from a structured web page. Web scraping was performed using the programming language Python with the help of google-play-scraper library. Data collection took place sometime in mid-April 2026, using the language filter in Indonesian (lang='id'), as well as the country set to Indonesia (country='id'). Data was then sorted using the newest sorting method (sort=Sort.NEWEST). This method was adopted since it allows efficient collection of data without any manual input process one after another. Data collection was performed on April 14–16, 2026, with the use of google-play-scraper library in Python. The scraping process applied the Indonesian language filter (lang='id') and Indonesia regional setting (country='id') to ensure that only Indonesian-language reviews from users in Indonesia were collected. The review retrieval process used the Sort.NEWEST parameter to obtain the most recent reviews available at the time of collection. Several review attributes were extracted, including user name, review content, rating score, review date, and thumbs-up count. A total of 4,891 reviews were successfully collected before preprocessing. During the cleaning stage, duplicate and empty reviews were removed, resulting in a final dataset of 4,790 reviews used for sentiment classification.

According to Busrayan and Andrianingsih [16], the web scraping method is the most effective technique in obtaining large-scale public data because it can run automatically, quickly, and consistently. In addition, Marganingsih et al. [17] Explains that the use of libraries such as Google-play-scraper allows the process of retrieving review data to be carried out in real-time, so that the data obtained is more relevant and in accordance with current conditions. This is reinforced by Jannah[18] who emphasize the importance of ensuring that the data collected is genuinely sourced from the original user so that the results of the analysis are not biased and remain scientifically valid.

2.2 Data Labeling

Once the data has been successfully collected, the next stage is the data labeling process which aims to determine the sentiment category of each user review. In this study, labeling was carried out using a heuristic approach (heuristic approach) based on user ratings (ratings) on a scale of 1 to 5. While this heuristic method allows automatic labeling on a large scale, it is important to acknowledge the limitation of this approach. Rating and review text may not always match due to user error, sarcasm, or complex opinions. Despite these minor discrepancies, converting high ratings as positive sentiment, medium as neutral, and low as negative remains a stable approach for building labeled datasets. [16].

In this study, labeling was carried out using a heuristic approach based on user ratings on a scale of 1 to 5. Wicaksono and Santi [19] stated that the use of ratings as ground truth is an objective method because it directly reflects the level of user satisfaction. This approach also reduces reliance on manual interpretations that are subjective.

Furthermore, Ningsih et al. [20] and Rafsanjani et al. [9] explains that converting high ratings as positive sentiment, medium ratings as neutral, and low ratings as negative are the most stable approaches in building labeled datasets. This method is considered to be able to describe the distribution of sentiment systematically and consistently.

2.3 Text Preprocessing Stage

After the labeling process is done, the next process is entering into text preprocessing. Text preprocessing is the process of cleaning, normalizing, and simplifying the text for processing. Digital review data usually consist of noise in the form of too much punctuation, variations in writing, and unnecessary words.

Some of the preprocessing steps performed include cleaning, case folding, tokenizing, stopword removal, and stemming. As stated by Ningsih [6] , the preprocessing step is an important step in text

analysis that helps in standardizing data for homogenization purposes. Also, according to Al-Husna et al. [2] case folding and cleaning have the potential to minimize word variation without any substantial meaning.

Tokenizing involves breaking down the text into word units. Stopword removal is aimed at removing frequent words without any high informational content. Lastly, the stemming process involves converting the word with suffix to the root word. Akbar et al. [6] stated that the utilization of Sastrawi stemmer is highly recommended in Indonesian language because it is highly efficient. [20],

2.4 TF-IDF Feature Extraction

Following this process, the textual data is transformed numerically by utilizing the Term Frequency-Inverse Document Frequency (TF-IDF). The TF-IDF technique is employed for weighting terms according to their significance within a document relative to the entire document in the data set.

TF-IDF, as per Susanto et al. [21], is a highly effective approach for analyzing text due to its ability to identify significant terms. The mathematical formulation of the TF-IDF formula, as mentioned by Kirana et al. [22], is as follows. [22]

Equation (1): TF-IDF

$$w_{(t,d)} = tf_{(t,d)} \times \log\left(\frac{N}{df_t}\right) \quad (1)$$

In this mathematical formulation, each symbol represents a specific component of the weighting scheme. The symbol $w_{(t.d)}$ represents the final mathematical weight of term t in document d. The component $tf_{(t.d)}$ denotes the term frequency, which represents the number of times term t specifically appears within the text of document d. The mathematical operator log refers to the logarithmic calculation performed on the inverse document frequency values. The variable N refers to the total number of review documents in the entire dataset. The variable $(df)^t$ refers to the frequency of documents, which include the particular term t in the dataset. Through the multiplication of these components, the algorithm achieves assigning of relatively high numerical values to terms that occur frequently in one document but infrequently in the entire dataset.

2.5 Splitting Data

Before entering the modeling stage, the dataset is divided into two parts, namely training data and test data with a ratio of 80:20. The goal of this sharing is to ensure the model has enough data to learn while remaining tested on data that has never been seen before.

According to Prasetyo et al. [23] proportional data sharing is essential to avoid overfitting problems, i.e. conditions when the model is over-tuned to the training data.

2.6 Support Vector Machine (SVM) Classification

The Support Vector Machine (SVM) algorithm is used in the construction of the classification model whereby the best separating hyperplane between sentiment classes is sought. According to Alkhoze and Almasre [10],, SVM is highly efficient in dealing with high-dimensional data like text due to its strong generalization capability.

The Support Vector Machine (SVM) model was constructed using a linear kernel in order to find the optimal hyperplane of sentiment classification. The regularization parameter was set at 1.0 to ensure a balance between the maximum margin and the minimum classification error. As a result of using a linear kernel, there was no need for gamma parameter in this case. There was no use of grid search optimization whereby fixed manual parameters were used for building the model. On the other hand, the Multinomial Naïve Bayes model was constructed using a default value of the smoothing parameter alpha = 1.0 (Laplace Smoothing) in order to ensure that there would be no zero probability. The data were split into training and test datasets by using the ratio 80:20. All models were built using the Scikit-learn library in Python. The optimization process is based on the following equations:

Equation (2): SVM Optimization

$$\min\left(\frac{1}{2}\| w \|^2 + C\sum_{i=1}^{n} \varepsilon_i\right) \quad (2)$$

In this mathematical formulation, each symbol represents a specific component of the algorithm. The notation "min" denotes the main objective of the algorithm which is to minimize the function. The fraction $\frac{1}{2}$ is a mathematical constant used to simplify the derivative calculation during the optimization process. The symbol "w" is the weight vector that determines the orientation of the hyperplane, while $||w||^2$ represents the squared Euclidean normvof the weight vector. Minimizing this norm ensures the maximization of the margin between classes.Furthermore, the symbol "C" acts as the regularization parameter set to 1.0 in this study which controls the penalty for classification errors. The symbol $\sum$ represents the summation operator, which adds up the values for all data points starting from index i up to the total number of data n. The symbol $\mathcal{E}_i$represents the slack variable which measures the error distance for data points that fall of the wrong side of the margin. Lastly, $x_i$denotes the feature vector of a specific review document, y_i is the actual true sentiment class label of that document, and b is the bias term that shifts the hyperplane away from the origin.

2.7 Naive Bayes Multinomial Classification

The Multinomial Naive Bayes algorithm was used for the comparison method, with an alpha value of 1.0. This means that Laplace smoothing is used in this model, which is crucial for dealing with unseen word features within the training data set, hence avoiding any zero-probability problems when doing the sentiment classification. The ratio between training and testing data is 80/20 respectively. These algorithms have been developed using Scikit-learn library in Python. Based on Apriyani and Kurniati [24], this algorithm determines the probability of word occurrence for a particular class using Bayes' Theorem.

The basic equations used are as follows:

Equations (3): Bayes Theorem

$$P(c \mid d) = \frac{P(d|c)\cdot P(c)}{P(d)} \quad (3)$$

In this fundamental equation, "P" stands for the probability in mathematics. "C" is the sentiment class, whereas "d" is the document or the textual review. Hence, "P(c|d)" is the posterior probability of the sentiment class when the particular document is given. "P(d|c)" is the likelihood of the document belonging to that particular sentiment class. "P©" is the prior probability or general frequency of the sentiment class, while "P(d)" is the marginal probability of the document belonging to the overall data.

2.8 Model Evaluation

The confusion matrix is the most comprehensive evaluation method because it is able to describe the model's performance in detail through indicators such as truelpositive, falsevpositive, true negative, and falsevnegative. In addition, Confusion matrix is very important in comparing algorithms because it is able to show the level of prediction error specifically. Kevin and Enjeli [25] also added that the combination of confusion matrix with Grid Search can help find the best parameters to improve model performance, especially in SVM algorithms which often show more stable results than Naive Bayes in some scenarios.

RESULTS AND DISCUSSION

3.1 Data Scraping

The scraping process was conducted using Python with the help of the **google-play-scraper** library to collect review data from the Google Play Store. The collected data includes user information, rating, review date, and original review text. here is a small part of the extraction data at table 1.

Table 1. Initial Data of Scraping Results

Review_Id	Username	Rating	Date	Original Review
db6cbd60-db87-4edb-bf14-23821b1331d7	sambusir yusuf	5	2026-04-12 13:39:14	ok.. good service
af75f2d0-bc49-4d68-a0b8-8b18fd72a0b8	Andhika Aditya Putra	5	2026-03-23 09:56:30	The error continues to be the application please fix it for ...
f54fc34e-3859-4038-b263-98455320cb3a	dwi satriawan	5	2026-03-19 17:51:28	Garuda is the best (service) flight attendant, b...
4faf4f8f-0711-49f9-abbe-a454987c31b6	Abdul Fattah	1	2026-03-16 03:50:21	When you want to print a boarding pass, even blank put...
5b6d8d87-d330-4b20-abce-caf05597ecd9	luthfi abdillah	2	2026-03-14 01:46:58	Booking above 9pax must be careful because...

The initial target of data collection was set at 26,000 reviews. However, after the scraping process was completed and the data underwent an initial validation stage, only 4,891 reviews were considered suitable for further processing. During the text preprocessing stage, which included text cleaning and the removal of symbols and punctuation, 101 reviews became empty and were therefore removed from the dataset. As a result, the final dataset used for feature extraction and classification consisted of 4,790 valid reviews.

3.2 Data Labeling (Data Labeling)

The data labeling step is done after the data collection procedure has been completed. For this study, The labels provide a foundation for the algorithm to learn the relationship pattern between the expected input and output features. Nonetheless, the data obtained through web scraping from the Google Play Store at an earlier phase is still considered raw data without sentiment labels. Thus, the data labeling step is done automatically using the rating feature (1 to 5-star rating) provided by users when leaving a comment. The assumption utilized in this step is that the rating reflects the user's satisfaction level toward the application's service. The labeling method adopted follows a heuristic rule that was systematically created as follows: [26]

1. Reviews with high ratings, i.e. 4 and 5, are automatically categorized as **positive** sentiment. This condition shows that the user has a good experience, feels satisfied, and assesses the airline's application or service running optimally. [28]
2. Reviews with a moderate rating, i.e. 3, are classified as neutral sentiment. In this category, users tend to give judgments that are in the middle, not showing strong satisfaction but also not expressing significant complaints. [29]
3. Reviews with low ratings, i.e. 1 and 2, are classified as negative sentiment. This category generally reflects dissatisfaction, technical glitches such as bugs in the application, or bad experiences experienced by users while using the service.[27]

This rule-based labeling method has proven to be very efficient in handling large amounts of data, as it is able to provide annotations automatically without having to involve manual processes by humans. This is important considering that manual labeling takes a very long time and has the potential to produce subjective bias between annotators. [30]

It is important to acknowledge the inherent limitations of this automated labeling approach. The primary limitation is the occasional mismatch between the user rating and their actual written review. For instance, a user might write a complaint about an application error but mistakenly leave a five star rating, which the system then automatically categorizes as positive. This discrepancy can occur due to user error, sarcasm, or mixed opinions within a single review. Despite these specific edge cases, relying on the user provided star ratings remains a highly efficient, objective, and widely accepted method for establishing ground truth labels in large scale datasets, operating on the assumption that the vast majority of users align their star ratings with their textual feedback.All data labeling is labeled through user comment ratings starting from a positive rating of 5,4 to a neutral rating of 3 and then towards a negative rating of 2 or 1,as shown in the example in table 2. [22].

Table 2. Examples of Data Labeling Results

Original review	Rating	Sentiment
ok.. good service	5	Positive
The error continues to be the application please fix it for …	5	Positive
Garuda is the best (service) flight attendant, b…	5	Positive
When you want to print a boarding pass, even blank put…	1	Negative
Booking above 9pax must be careful because…	2	Negative
The app is very bad. Redeem miles that are not…	1	Negative
ok.. ok	5	Positive
Live Chat New Error	1	Negative
Beta version is awaited	5	Positive
Regression. Checkin from the app is directed to…	1	Negative

Data distribution is carried out to provide a more intuitive picture of the composition of the sentiment classes in the dataset. After the data cleaning process, the final dataset of 4,790 reviews shows the following distribution: the dataset is dominated by positive sentiment with 3,075 reviews (64.20%). Negative sentiment consists of 1,310 reviews (27.35%), and neutral sentiment is the minority class with 405 reviews (8.45%).

The results indicate that the neutral class achieved the lowest performance due to class imbalance, where neutral reviews only represented 8.45% of the dataset. Below the table, a visualization of the figure 2.

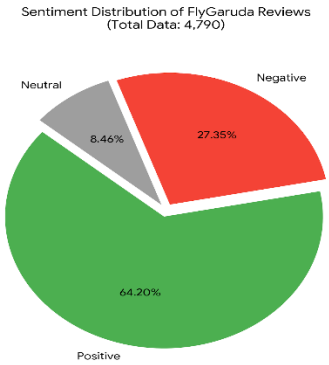

Figure 2. Examples of Sentiment Labeling Results

The number of labeled datasets is 4,891 reviews. Further, the dataset may be segregated as train and test datasets based on an 80:20 ratio by using Stratified Sampling, which ensures that the percentage of samples in each sentiment class stays the same. This is necessary to minimize the chances of model bias for the majority class. [16].

3.3 Pre-Processing Stages (Text Preprocessing)

Data obtained from social media or digital platforms usually have various types of noises such as symbols, different writing styles, and inconsistency in sentence structure.

This phase is critical since the data quality has a significant impact on the performance of the classification algorithm. The process of text pre-processing in this research consists of several major sub-phases, which include the following.

3.3.1 Case Folding dan Cleaning

Firstly, the process of case folding occurs when all the data are converted to lowercase (lowercase). It is done so that the system ignores differences in letter casing, and words such as “Garuda”, “GARUDA”, and “garuda” would be treated as a single word.

After going through the case folding phase, a cleaning process took place with Regular Expression (Regex) using Python by eliminating unnecessary symbols including numbers (\d+), punctuation mark ([^\w\s]), emojis, other special symbols, extra spaces, and other irrelevant components. Even though the scraping parameter was set to use the Indonesian language (lang=’id’), some of the scraped reviews were written in a combination of Indonesian, English language, and casual language that people use when giving feedback in applications. Thus, the cleaning phase was aimed at keeping only valid letters for the subsequent tokenization stage. The removal of these elements was intended to reduce noise in the dataset and improve the effectiveness of sentiment analysis.

3.3.2 Tokenization

The subsequent step is tokenization, whereby the text is divided into units known as tokens. Tokenization is accomplished by the use of the word_tokenize function of the Natural Language Toolkit (NLTK).

The result of this process is a change in the data structure from a whole sentence to a separate list of words. The output of this stage is represented in the form of a data list structure with square brackets [ ], which indicates that each sentence has been converted into a set of individually processable tokens.

3.3.3 Stopword Removal

Stopwords are common words in language that have little or no significance in sentiment analysis. Examples of stopwords in Indonesian include "yang", "and", "di", "ke", "from", and "ini".

The collected reviews were primarily written in Indonesian, although several reviews also contained English terms and informal slang expressions. For stopword removal, this study used the Indonesian stopword list provided by the NLTK corpus to automatically eliminate common words with little semantic significance in sentiment analysis. Furthermore, the stemming process was conducted using the Sastrawi library in Python to convert inflected words into their root forms.

3.3.4 Stemming

The last stage in preprocessing is stemming, which is the process of changing the word with suffix into its basic form. In Indonesian, a single word can have many variations of forms such as "repair", "repair", and "repair".

If not normalized, the three words will be considered different features by the model. Therefore, the Sastrawi library is used with the StemmerFactory module to eliminate suffixes and return words to their basic form, for example to be "good".The stemming process is very important in the analysis of Indonesian texts because it is able to significantly reduce the dimension of features while unifying the meaning of cognate words.

After all stages of preprocessing are completed, including the removal of empty data due to the cleaning process, the number of final data ready to be used in the analysis process is reduced to 4,790 rows of data. All preprocessing results were taken from only 5 examples and are shown in figure 3.

	Content	Case Folding	Cleaning	Tokenization	Stopword Removal	Stemming
0	ok.. good service	ok.. good service	ok good service	['ok', 'good', 'service']	['ok', 'good', 'service']	['ok', 'good', 'service']
1	The application keeps erroring, please fix it for...	the application keeps erroring, please fix it for ...	the application keeps erroring please fix it for ...	['the', 'application', 'keeps', 'erroring', 'please', 'fix', 'it', 'for', ...	['application', 'keeps', 'erroring', 'please', 'fix', ...	['applic', 'keep', 'error', 'pleas', 'fix', ...
2	Garuda is the best (its service) flight attendants, s...	garuda is the best (its service) flight attendants, stor...	garuda is the best its service flight attendants stor...	['garuda', 'is', 'the', 'best', 'its', 'service', 'flight', 'attendants', ...	['garuda', 'best', 'service', 'flight', 'attendants', ...	['garuda', 'best', 'servic', 'flight', 'attendant', ...
3	When I wanted to print the boarding pass, it went blank put...	when i wanted to print the boarding pass, it went blank whi..	when i wanted to print the boarding pass it went blank whi..	['when', 'i', 'wanted', 'to', 'print', 'the', 'boarding', 'pass', ...	['wanted', 'print', 'boarding', 'pass', ...	['want', 'print', 'board', 'pass', ...
4	booking above 9pax must be careful because...	booking above 9pax must be careful because..	booking above pax must be careful because..	['booking', 'above', 'pax', 'must', 'be', 'careful', 'because', ...	['booking', 'pax', 'careful', ...	['book', 'pax', 'care', ...

Figure 3. Sample Review After PreProcessing

3.4 TF-IDF Feature Extraction Results

Before transforming the data into numerical form, this study first conducted an initial exploration of word characteristics using Word Cloud visualization. This technique is used to give a preliminary idea of the words that appear most often in user reviews, both in positive and negative categories. All the most frequently occurring words will be shown as in figure 4.

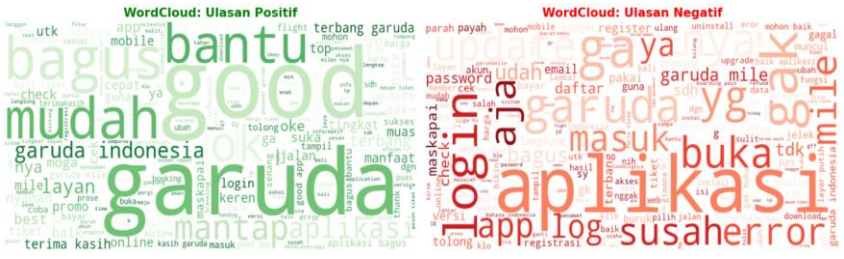

Figure 4. Positive Word Cloud and Negative Word Cloud Visualizations

Based on the results of the visualization, it can be seen that there is a difference in the word pattern used between positive and negative reviews. Positive reviews tend to be dominated by words like "good," "good," "steady," and "easy," while negative reviews contain more words like "error," "bad," and "can't."

Once visual inspection of the data has been done, the data will be subjected to further processing through the use of the term frequency-inverse document frequency (TF-IDF) feature extraction technique. The purpose of this technique is to transform the text data into numerical form by assigning weights to each word depending on how frequently it occurs in a document and also how rarely it appears in the data set. This process creates a feature matrix whose dimensions are (4790, 3911), which means that there are 4,790 clean review documents and 3,911 words that have been extracted from the entire data set.

As an initial representation, the ten words with the highest TF-IDF weights in the dataset that have the highest weighted values generally represent words that appear frequently and have a high level of relevance to the content of the review. Words such as "good", "nice", and "stable" tend to appear in positive reviews, while words such as "application" and "garuda" are common but still contribute to the formation of text patterns. The representation is shown in Table 3.

Table 3. Ten Words with the Highest TF-IDF Weight

Cover	Total Bobot TF-IDF	Percentage Weight (%)
good	263.258612	16.344780
good	214.578257	13.322392
People	178.909209	11.107829
ok	176.193679	10.939231
Garuda	172.872916	10.733057
Application	170.188606	10.566398
Steadfast	152.714082	9.481468
Easy	130.716285	8.115704
Indonesia	81.188856	5.040724
Good	70.038174	4.348418

3.5 Data Splitting

The next step for this research is the splitting of data into training data and test data. The separation is done to make sure that the model can be trained and tested separately. As the result, the performance will become objective and unbiased.

The labeled and preprocessed data will be separated using the split ratio of 80% for the training dataset and 20% for the testing data. This separation produces 3,832 data for model training and 958 data for testing. The distribution of data can be viewed in the pie chart on figure 5.

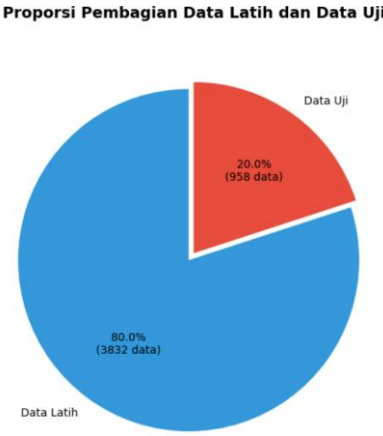

Figure 5. Pie Chart Visualization of Data Sharing Proportions

This data sharing is also carried out using the Stratified Sampling technique, which aims to maintain a balance of class distribution in both data subsets. With this approach, the proportion of sentiment classes remains consistent so that the model can learn more fairly to each category.

3.6 Results of Training and Evaluation of the Support Vector Machine (SVM) Model

The first algorithm analyzed in this research is the Support Vector Machine (SVM), which acts as the main model in the user review classification process for the FlyGaruda app. When considering the Support Vector Machine (SVM) and the Multinomial Naïve Bayes algorithm, the research considers two main criteria: performance metrics and computational efficiency. For the experiment to be replicable, all the model settings were tailored based on the observations made during training the data. In particular, the SVM model was trained with a linear kernel and C=1.0 as a regularizer parameter; there was no gamma or grid search as a result of the linear kernel selection. The Multinomial Naïve Bayes model was trained with a smoothing parameter (Laplace smoothing) set to α=1.0. Performance metrics of each model will be assessed based on the Confusion Matrix with such metrics as Accuracy, Precision, Recall, and F1-Score. This measure is chosen as it determines how effectively each model handles imbalanced classes. Also, computational efficiency will be measured based on the amount of time each algorithm spends training the high-dimensional TF-IDF feature matrix. Thus, SVM and Multinomial Naïve Bayes models will be compared in terms of both classification effectiveness and computational efficiency when using. Therefore, the comparison between SVM and Multinomial Naive Bayes in this study focuses on classification effectiveness and computational efficiency using the same TF-IDF feature representation and train-test data split.In the training stage, the SVM model recorded a computation time or training time of 90.6048 seconds. The results of the SVM performance model are shown in Figure 6.

```
=== PERFORMA MODEL SVM ===
              precision    recall  f1-score   support

     negatif       0.73      0.80      0.76       262
      netral       0.25      0.01      0.02        81
     positif       0.87      0.94      0.90       615

    accuracy                           0.82       958
   macro avg       0.61      0.58      0.56       958
weighted avg       0.78      0.82      0.79       958

--- Metrik Spesifik SVM ---
SVM Accuracy  : 0.8225469728601252
SVM Precision : 0.776600162653174
SVM Recall    : 0.8225469728601252
SVM F1-Score  : 0.7890855536313094

Confusion Matrix Mentah (SVM):
[[209   2  51]
 [ 42   1  38]
 [ 36   1 578]]
```

Figure 6. SVM Model Performance

As per the results provided in the image, it can be inferred that the SVM classifier has quite a good capability to detect patterns related to sentiments present in the test review data. A total of 958 test review data was used as input to the model, and out of them, the model could correctly classify more than 82% of the data using proper labeling, thereby giving an accuracy score of 82.25%.

In the image above, evaluation measures for more detailed evaluation metrics have been performed. The SVM classifier has scored a precision rate of 0.7766 or 77.66%. This means that the accuracy of prediction made by the model concerning the positive class of the data is quite high. The recall rate of 0.8225 or 82.25% tells us that the classifier has the potential to capture most of the relevant data in every class.

3.7 Performance Evaluation Results of the Naive Bayes Model

In the next stage, this study uses the Multinomial Naive Bayes algorithm as a comparative model for SVM. This model is applied to the same feature matrix resulting from the TF-IDF process, so that the comparisons made are fair and consistent.

At the training stage, the Multinomial Naive Bayes algorithm takes only about 0.0802 seconds, which shows very high computational efficiency.Then the results of the naive Bayes performance model are shown in Figure 7.

```
=== PERFORMA MODEL NAIVE BAYES ===
              precision    recall  f1-score   support

     negatif       0.74      0.77      0.76       262
      netral       0.00      0.00      0.00        81
     positif       0.85      0.95      0.90       615

    accuracy                           0.82       958
   macro avg       0.53      0.57      0.55       958
weighted avg       0.75      0.82      0.79       958

--- Metrik Spesifik Naive Bayes ---
MultinomialNB Accuracy  : 0.8225469728601252
MultinomialNB Precision : 0.7514857866797181
MultinomialNB Recall    : 0.8225469728601252
MultinomialNB F1-Score  : 0.7852162975904272

Confusion Matrix Mentah (Naive Bayes):
[[202   0  60]
 [ 41   0  40]
 [ 29   0 586]]
```

Figure 7. Performa Model Naïve Bayes

The test results show that the **Naive Bayes algorithm** achieved an accuracy of **82.25%**, the same as SVM. However, class-based evaluation shows differences in performance.

Naive Bayes obtained a **precision of 75.15%**, **recall of 82.25%**, and **F1-score of 78.52%**. Although the recall is high, the lower precision indicates that this model produces more prediction errors than SVM.

Further analysis shows that Naive Bayes has difficulty recognizing the **neutral class**, with precision and recall values of **0.00**. This indicates model bias toward the majority classes, where neutral data is often classified as positive or negative.

3.8 Model Comparison Evaluation

In order to continue with this study, the next step is to make a comparative analysis between both SVM and Multinomial Naïve Bayes in terms of their efficiency in sentiment classification. The first method used in this case was the confusion matrix, which is used to measure the distribution of the prediction of the model in relation to the real class, so that it will be easier to analyze the distribution of the classification errors made by the system. Model comparison of the confusion matrix of the 2 algorithms can be seen in Figure 8 below.

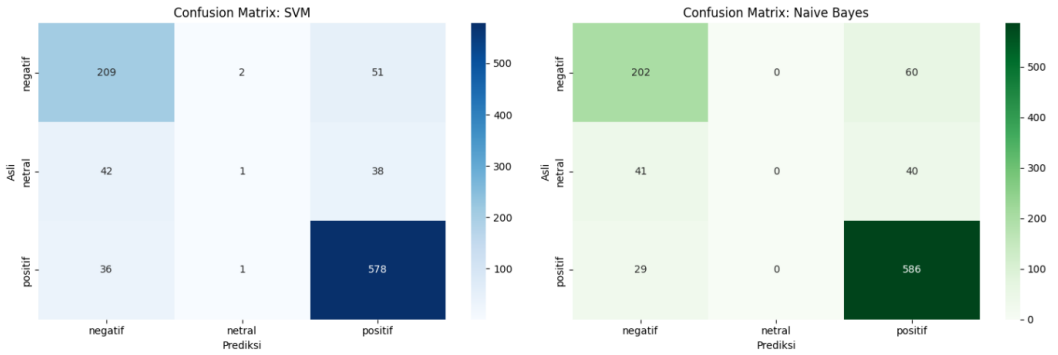

Figure 8. Visualization of the SVM and Naïve Bayes Confusion Matrix

From the result of the confusion matrix above, it can be seen that the SVM model makes more accurate predictions, since it creates a more stable and balanced distribution of predictions. Mostly, the predictions were found in the main diagonal area, which means most of the predictions were correctly classified.

In contrast, the Naive Bayes model shows a more pronounced pattern of imbalances. Although it is quite good at classifying positive classes, it fails to accurately detect neutral classes, which is evident from the lack of values on the main diagonal for that class. This indicates that the model is more likely to ignore minority classes and distribute them to other classes. The contrasting differences are made into a visual bar chart in figure 9.

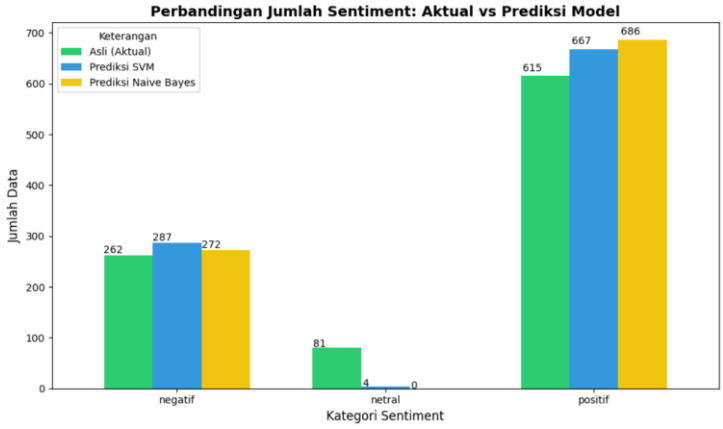

Figure 9. Visualization Actual vs prediction comparison bar chart

The comparison bar chart reinforces the previous finding that although the two models have similar accuracy values, the distribution of prediction errors in each class shows significant differences. SVM looks more consistent in maintaining a balance of predictions between classes, whereas Naive Bayes shows a biased tendency towards dominant classes.

As for the evaluation results, it is clear that the two algorithms achieve the same accuracy value of 82.25%; however, the quality of predictions varies among the different classes. For instance, the SVM model has high cross-class prediction stability, having a weighted precision of 77.66% and an F1 score of 78.91%. On the other hand, Naïve Bayes' weighted precision is 75.15% with an F1 score of 78.52%. This can be observed in the comparison of the model evaluations in Table 4.

Table 4. Model Evaluation Comparison Results

Algoritma	Akurasi	Presisi (Weighted)	Recall (Weighted)	F1-Score (Weighted)	Training Time (s)
SVM	82,25%	77,66%	82,25%	78,91%	90.604787
Naive Bayes	82,25%	75,15%	82,25%	78,52%	0.080212

In a more in-depth analysis of the class-wise prediction results, it is evident that there is a major difference in performance in relation to handling imbalanced sentiment data. As such, while the SVM model succeeded in classifying the three sentiment classes, namely positive, negative, and neutral sentiments, the Multinomial Naïve Bayes algorithm did not predict the neutral class, resulting in precision and recall values of zero. This means that the SVM model works better on imbalanced sentiment data.

Table 5. Class-wise Evaluation Metrics

Algoritma (class)	Precision	Recall	F1-Score
SVM (neutral)	0.25	0.01	0.02
Naive Bayes (neutral)	0.00	0.00	0.00
SVM (Weighted Avg)	0.78	0.82	0.79
Naive Bayes (Weighted Avg)	0.76	0.82	0.79

CONCLUSION

This study conducted a sentiment analysislof userlreviews for the FlyGaruda applicationvon the Google Play Store using SupportlVectorlMachine (SVM) and Multinomial Naïve Bayes algorithms, with TF-IDF used as the featuresextractionsmethod. The experimental results show that both models achieved the same overall accuracy of 82.25%. However, differences were observed in classification balance and computational efficiency. The Support Vector Machine model achieved slightly higher weighted precision (77.66%) and F1-score (78.91%), and showed better performance in handling the minority neutral class based on the class-wise evaluation results. Meanwhile, the Multinomial Naïve Bayes model demonstrated significantly faster computational performance, requiring only 0.08 seconds of training time compared to 90.60 seconds for the SVM model. These findingsmindicate that SVM is more suitablenfor balanced sentiment classification performance, whereas Naïve Bayes is more efficient for rapid computation and lightweight implementation. Future research may compare these methods with other machine learning or deepnlearningnapproaches, such as KNN or IndoBERT, to further improve sentiment classification performance.

REFERENCES

[1] M. Rizqi, Martanto, A. R. Dikanda, and D. Rohman, "Naive Bayes Algorithm to Enhance Sentiment Analysis of Coursera Application Reviews on Google Play Store," *J. Artif. Intell. Eng. Appl.*, vol. 4, no. 2, pp. 823–829, 2025, doi: 10.59934/jaiea.v4i2.758.

[2] G. S. Al-Husna, D. Asmarajati, I. A. Ihsannuddin, and R. Mahmudati, "Perbandingan Metode Naïve Bayes dan Support Vector Machine untuk Analisis Sentimen pada Ulasan Pengguna Aplikasi LinkedIn," *STORAGE J. Ilm. Tek. dan Ilmu Komput.*, vol. 3, no. 2, pp. 139–144, 2024, doi: 10.55123/storage.v3i2.3602.

[3] Y. A. Budi Harijanto and L. Miftahurroifa, "Penerapan Algoritma Naïve Bayes untuk Klasifikasi Retensi Arsip," *J. Inform. Polinema*, vol. 4, no. 2, p. 155, 2018, doi: 10.33795/jip.v4i2.159.

[4] J. O. Leandro and M. I. Fianty, "Evaluation of Sentiment Analysis Methods for Social Media Applications: A Comparison of Support Vector Machines and Naïve Bayes," *Int. J. Informatics Vis.*, vol. 9, no. 2, pp. 796–807, 2025, doi: 10.62527/joiv.9.2.2905.

[5] Alfandi Safira and F. N. Hasan, "Analisis Sentimen Masyarakat Terhadap Paylater Menggunakan Metode Naive Bayes Clasifier," *Zo. J. Sist. Inf.*, vol. 5, no. 1, pp. 59–70, Jan. 2023, doi: 10.31849/zn.v5i1.12856.

[6] M. R. Akbar, S. Defit, and Sumijan, "Metode Support Vector Machine dan Naïve Bayes untuk analisis sentimen Ibu Kota Nusantara," *J. KomtekInfo*, vol. 11, no. 4, pp. 323–331, 2024, doi: 10.35134/komtekinfo.v11i4.579.

[7] Satrio Junaidi, R. Valicia Anggela, and D. Kariman, "Klasifikasi Metode Data Mining untuk Prediksi Kelulusan Tepat Waktu Mahasiswa dengan Algoritma Naïve Bayes, Random Forest, Support Vector Machine (SVM) dan Artificial Neural Nerwork (ANN)," *J. Appl. Comput. Sci. Technol.*, vol. 5, no. 1, pp. 109–119, 2024, doi: 10.52158/jacost.v5i1.489.

[8] R. Rakarahayu Putri and N. Cahyono, "Analisis Sentimen Komentar Masyarakat Terhadap Pelayanan Publik Pemerintah DKI Jakarta Dengan Algoritma Super Vector Machine dan Naive Bayes," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 8, no. 2, pp. 2363–2371, Apr. 2024, doi: 10.36040/jati.v8i2.9472.

[9] A. S. Agil Rafsanjani, D. L. Fithri, and S. Supriyono, "Sentiment Analysis of User Reviews of the KitaLulus Application on Google Play Store using the Support Vector Machine (SVM) Algorithm," *Sistemasi*, vol. 14, no. 5, p. 2519, 2025, doi: 10.32520/stmsi.v14i5.5519.

[10] M. Alkhoze, "Sentiment Analysis of Mobile Legends Play Store Reviews Using Support Vector Machine and Naive Bayes," *J. Digit. Mark. Digit. Curr.*, vol. 2, no. 4, pp. 368–389, 2025, doi: 10.47738/jdmdc.v2i4.44.

[11] B. A. Maulana, M. J. Fahmi, A. M. Imran, and N. Hidayati, "Analisis Sentimen Terhadap Aplikasi Pluang Menggunakan Algoritma Naive Bayes dan Support Vector Machine (SVM)," *MALCOM Indones. J. Mach. Learn. Comput. Sci.*, vol. 4, no. 2, pp. 375–384, 2024, doi: 10.57152/malcom.v4i2.1206.

[12] M. A. Java, Mohammad Syafrullah, W. Windarto, and P. Painem, "Analisis Sentimen Ulasan Pengguna Aplikasi Threads pada Google Play Store Menggunakan Multinomial Naive Bayes dan Support Vector Machine," *J. Ticom Technol. Inf. Commun.*, vol. 12, no. 2, pp. 75–80, 2024, doi: 10.70309/ticom.v12i2.112.

[13] B. BAS, "Perbandingan Kinerja Model Support Vector Machine dan Naive Bayes Untuk Analisis Sentimen Super App Polri," *Rabit J. Teknol. dan Sist. Inf. Univrab*, vol. 11, no. 1, pp. 1962–1976, Feb. 2026, doi: 10.36341/rabit.v11i1.7582.

[14] M. F. Baihaqi, L. Magdalena, and R. Fahrudin, "Analisis Sentimen Aplikasi Deepseek Menggunakan Metode Naive Bayes dan Support Vector Machine," *RIGGS J. Artif. Intell. Digit. Bus.*, vol. 4, no. 3, pp. 4051–4062, 2025, doi: 10.31004/riggs.v4i3.2511.

[15] M. Bayu Anggara, "Comparison of Naïve Bayes and SVM Methods in Sentiment Analysis of User Reviews on the RSUD AL IHSAN Mobile Application," *Competitive*, vol. 20, no. 1, pp. 32–42, Jun. 2025, doi: 10.36618/competitive.v20i1.4213.

[16] O. M. Nurfauzi, S. S. Hilabi, F. Nurapriani, and B. Huda, "Analisis Sentimen Grab Indonesia pada Ulasan Google Play Store menggunakan Algoritma Naïve Bayes dan Support Vector Machine," SMARTICS Journal, vol. 11, no. 1, pp. 8–13, 2025.DOI: 10.21067/smartics.v11i1.11789

[17] D. Marganingsih, H. Oktavianto, and G. Abdurrahman, "Analisis Sentimen Komentar Youtube Masterchef Indonesia Menggunakan Algoritma Support Vector Machine dan Gaussian Naïve Bayes," *J. Inform. dan Teknol. Pendidik.*, vol. 5, no. 1, pp. 16–26, 2025, doi: 10.59395/jitp.v5i1.117.

[18] N. Z. B. Jannah and K. Kusnawi, "Comparison of Naïve Bayes and SVM in Sentiment Analysis of Product Reviews on Marketplaces," *Sinkron*, vol. 8, no. 2, pp. 727–733, 2024, doi: 10.33395/sinkron.v8i2.13559.

[19] A. Faiq Wicaksono and R. Candra Noor Santi, "Perbandingan Algoritma Naive Bayes dan Svm Terhadap Analisis Sentimen Pengguna Aplikasi Fatsecret Pada Google Play Store," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 10, no. 1, pp. 1074–1080, Jan. 2026, doi: 10.36040/jati.v10i1.16943.

[20] W. Ningsih, B. Alfianda, R. Rahmaddeni, and D. Wulandari, "Perbandingan Algoritma SVM dan Naïve Bayes dalam Analisis Sentimen Twitter pada Penggunaan Mobil Listrik di Indonesia," *MALCOM Indones. J. Mach. Learn. Comput. Sci.*, vol. 4, no. 2, pp. 556–562, 2024, doi: 10.57152/malcom.v4i2.1253.

[21] L. A. Susanto, "Komparasi Model Support Vector Machine Dan K-Nearest Neighbor Pada Analisis Sentimen Aplikasi Polri Super App," *J. Inform. dan Tek. Elektro Terap.*, vol. 12, no. 2, 2024, doi: 10.23960/jitet.v12i2.4152.

[22] K. Khalid, R. Wijaya, and M. A. Bijaksana, "Comparative Sentiment Analysis of Sirekap Application Reviews Using Support Vector Machines and Naive Bayes," INTEK: Jurnal Penelitian, vol. 12, no. 1, 2025. DOI: 10.31963/intek.v12i1.5196

[23] Y. A. Prasetyo, E. Utami, and A. Yaqin, "Pengaruh Komposisi Split Data Terhadap Performa Akurasi Analisis Sentimen Algoritma Naïve Bayes dan SVM," *J. Electr. Eng. Comput.*, vol. 6, no. 2, pp. 382–390, 2024, doi: 10.33650/jeecom.v6i2.9188.

[24] H. Apriyani and K. Kurniati, "Perbandingan Metode Naïve Bayes Dan Support Vector Machine Dalam Klasifikasi Penyakit Diabetes Melitus," *J. Inf. Technol. Ampera*, vol. 1, no. 3, pp. 133–143, 2020, doi: 10.51519/journalita.volume1.isssue3.year2020.page133-143.

[25] K. Kevin, M. Enjeli, and A. Wijaya, "Analisis Sentimen Pengunaaan Aplikasi Kinemaster Menggunakan Metode Naive Bayes," *J. Ilm. Comput. Sci.*, vol. 2, no. 2, pp. 89–98, 2024, doi: 10.58602/jics.v2i2.24

[26] Singgih Jatmiko & Charles Dometian, "Analisis Sentimen Ulasan Google Play Store: Studi Komparatif Algoritma SVM, Naïve Bayes, dan Logistic Regression," JURNAL FASILKOM, vol. 15, no. 3, 2025. DOI: https://doi.org/10.37859/jf.v15i3.10016

[27] Khalid, Rifki Wijaya, & Moch Arif Bijaksana, "Comparative Sentiment Analysis of Sirekap Application Reviews Using Support Vector Machines and Naïve Bayes," INTEK: Jurnal Penelitian, vol. 12, no. 1, 2025. DOI: https://doi.org/10.31963/intek.v12i1.5196

[28] Sebastianus Adi Santoso Mola, et al. Analisis Sentimen Aplikasi Halo BCA di Google Play Store Menggunakan Metode Naive Bayes, Support Vector Machine dan Random Forest. 2024. DOI: 10.52972/hoaq.vol15no2.p69-79.

[29] Bahtiar, S. A., Dewa, C. K., & Luthfi, A. Comparison of Naïve Bayes and Logistic Regression in Sentiment Analysis on Marketplace Reviews Using Rating-Based Labeling.Teknosi Jurnal. 2023. DOI: 10.25077/TEKNOSI.v11i1.2025.87-97

[30] Agus Nursikuwagus, et al. (2025). Support Vector Machine to Classify Sentiment Reviews on Google Play Store. BITS. DOI: 10.47065/bits.v7i3.8697