

Sentiment Classification of Telecommunication Application Reviews Using Naive Bayes and Support Vector Machine: A Comparison of TF-IDF Unigram and N-Gram Feature Representations

Karin Nisrina Andriyani¹, Hafiyyan Putra Pratama²

Department of Telecommunication Systems, Universitas Pendidikan Indonesia
karinnisrina73@upi.edu¹, hafiyyan@upi.edu²

Article Info

Article history:

Received 2026-08-27

Revised 2026-10-09

Accepted 2026-10-19

Keyword:

*MyTelkomsel,
Naive Bayes,
N-Gram,
Support Vector Machine,
TF-IDF*

ABSTRACT

Important information regarding user satisfaction and complaints about an application can be obtained from reviews uploaded to the Google Play Store. Telecommunications service applications are among the most widely used applications in Indonesia and receive numerous reviews that can be leveraged to evaluate service quality. However, manually analyzing large volumes of reviews is inefficient. This study aims to classify sentiment in telecommunication application user reviews by comparing TF-IDF Unigram (Baseline) and N-Gram (Unigram-Bigram) feature representations and evaluating Naive Bayes and Support Vector Machine (SVM), with hyperparameter validation using Grid Search applied separately to each feature scenario. The dataset consists of 2,836 user reviews collected from the Google Play Store through web scraping. The data underwent preprocessing, rating-based sentiment labeling with rule-based label correction, data splitting, and TF-IDF feature extraction for both scenarios before classification. Model performance was evaluated using accuracy, precision, recall, F1-score, F1-score macro average, and confusion matrix. The results show that SVM outperforms Naive Bayes in both feature scenarios. The best performance was achieved by SVM using the Baseline TF-IDF Unigram representation, with 92.43% accuracy, 91.98% precision, 88.24% recall, 90.07% F1-score, and 91.98% F1-score macro average. Grid Search produced identical optimal configurations for both scenarios, namely linear kernel, C=1, and gamma=scale, indicating their suitability for both feature representations. These findings indicate that, within the dataset and experimental configuration used, adding N-Gram features does not necessarily improve classification performance, particularly for SVM, while unigram representation alone effectively classifies telecommunication application user reviews.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.

I. INTRODUCTION

The improvement of application-based service quality has become a top priority in the telecommunications sector amid rapid digitalization. One of the widely used digital service applications in Indonesia is a telecommunication application that provides various services such as internet package purchases, balance and quota checking, bill payment, and customer service. As an application used by millions of users, this telecommunication application receives numerous reviews via the Google Play Store

containing various opinions, experiences, criticisms, and suggestions regarding the services provided. This information can be used as evaluation material to improve service quality. However, the very large number of reviews makes manual analysis less effective and quite time-consuming.

The application of Natural Language Processing (NLP) and text mining in the form of sentiment analysis serves to identify, group, and analyze user opinions toward a product or service. Through sentiment analysis, user opinions can be classified into positive and negative categories, thereby

facilitating the automatic evaluation process. Consequently, sentiment analysis is widely used in various studies to process large volumes of available user review data.

Several previous studies indicate that the use of the Support Vector Machine (SVM) algorithm achieves high performance and effectiveness in classifying sentiment in MyTelkomsel application reviews. A previous study reported that SVM achieved an accuracy of 83% on a dataset labeled using an expert-based approach [1]. Another study showed that SVM obtained an accuracy of 93.36% in sentiment classification of MyTelkomsel reviews [2]. In addition, a study comparing Naive Bayes and SVM on reviews of a mobile service provider application showed that SVM achieved better performance than Naive Bayes on MyTelkomsel review data [3]. These results indicate that both algorithms have good potential for use in sentiment classification tasks. Similar findings have also been reported for banking application reviews [4], digital wallet applications [5], social security applications [6], and social media application reviews on the Google Play Store [7], which consistently show that both the SVM and Naive Bayes algorithms are effective for classifying user review sentiment.

Nevertheless, previous studies still have several limitations. Most studies in the domain of sentiment analysis of application reviews still use TF-IDF unigram feature representation with default parameters, without testing N-Gram representation or performing systematic hyperparameter validation. For example, a study comparing several classification algorithms on application reviews used only a single feature representation without hyperparameter validation, and without addressing class imbalance in the dataset used [8]. Similarly, a study comparing four classification algorithms on product reviews also relied solely on a single TF-IDF representation without testing N-Gram combinations or parameter tuning [9].

Another study that tested several N-Gram configurations (unigram, bigram, and their combination) on the Naive Bayes and SVM algorithms was also not accompanied by hyperparameter validation, so the optimal performance of each feature representation configuration has not been fully explored [10]. In addition to feature representation, the effectiveness of Grid Search as a hyperparameter validation method is itself not necessarily consistent across various data conditions. One study on the SVM algorithm found that hyperparameter tuning does not always have an impact on improving performance, particularly on data that is linearly separable [11], while the application of Grid Search is generally carried out for various machine learning algorithms separately without linking it to a comparison of feature representations [12], [13].

Based on these limitations, this study offers novelty through a systematic comparison of TF-IDF Unigram and N-Gram feature representations, a performance comparison between Naive Bayes and Support Vector Machine on a telecommunication application review dataset, and the

application of Grid Search as a fair hyperparameter validation method across both feature scenarios. In addition, rule-based label correction was applied to reviews with mismatched rating-content sentiment to improve labeling quality.

Naive Bayes and SVM were selected as the comparison algorithms in this study based on their consistent use and reported effectiveness in prior sentiment analysis research on application reviews [1], [2], [3]. Naive Bayes offers computational efficiency and simplicity when handling high-dimensional TF-IDF features, while SVM is well known for its strong generalization capability in high-dimensional, sparse text data through margin-based hyperplane separation. Their frequent adoption as baseline algorithms in this research domain allows the findings of this study to be directly compared with existing literature addressing similar objects of study and data characteristics, thereby strengthening the validity and relevance of the resulting performance comparison.

Based on the above discussion, this study aims to classify the sentiment of telecommunication application reviews by comparing TF-IDF Unigram and N-Gram feature representations and evaluating the performance of Naive Bayes and SVM, with hyperparameter optimization via Grid Search applied separately to each feature scenario on the algorithm with the best results. This approach is intended to determine which feature representation more effectively captures sentiment signals within informal, user-generated Indonesian text, while ensuring that the selected model configuration is systematically validated rather than relying solely on default parameters. The resulting findings are expected to provide meaningful insight into user perception of the telecommunication application and to serve as a practical reference for future service quality improvement efforts.

II. METHOD

A. Research Workflow

The research was conducted systematically, starting from data collection to model result comparison, as illustrated in Figure 1.

B. Data Collection

The initial stage of this study began with the collection of MyTelkomsel application user review data available on the Google Play Store. Data collection was carried out using the Google Play Scraper library run through Google Colaboratory. The data obtained included ratings, review content, review dates, and other supporting information available on the application page. The scraping process produced a total of 3,000 MyTelkomsel application user reviews, with review dates spanning a nine-day period, from May 6, 2026, to May 15, 2026. The distribution of the collected reviews across rating values was strongly bimodal, with 1,461 reviews rated 1, 208 reviews rated 2, 164 reviews

rated 3, 161 reviews rated 4, and 1,006 reviews rated 5, totaling 3,000 reviews. This rating distribution is consistent with the class imbalance later observed in the labeled dataset, which consisted of 1,732 negative and 1,104 positive reviews.

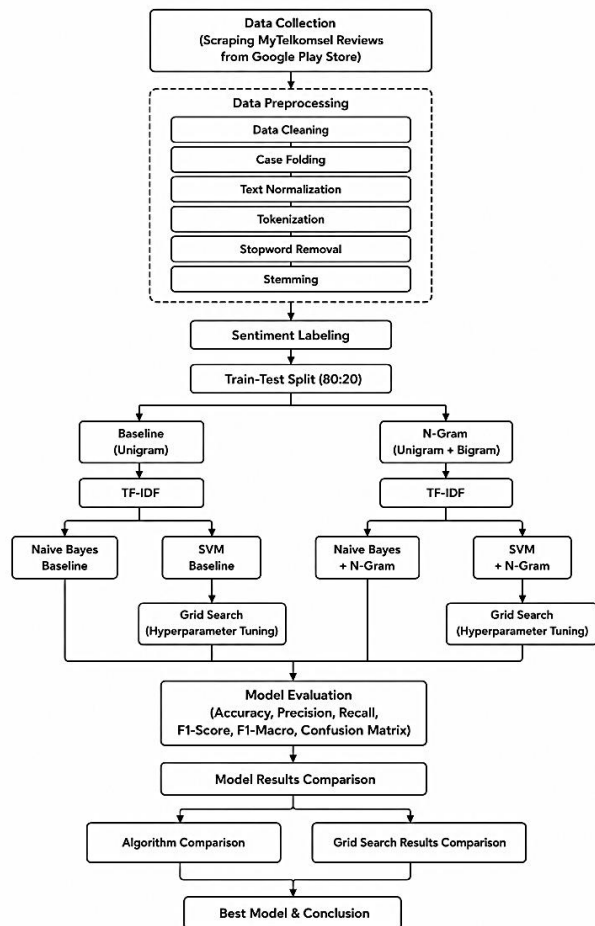


Figure 1 MyTelkomsel Sentiment Classification Workflow

As part of the data quality check, a duplicate-review check was also conducted on the raw dataset. The results showed no fully identical rows when considering the combination of username, rating, review text, and date together. However, 506 rows were found to contain identical review text originating from different users; this is a pattern commonly observed in short and generic reviews (e.g., "bagus" or "lemot"), rather than an indication of scraping errors or spam activity.

TABEL I
EXAMPLE OF MYTELKOMSEL USER REVIEW SCRAPING DATA RESULTS

Username	Rating	Review	Date
Pengguna Google	1	jaringan sangat lelet, harga paket mahal2	2026-05-15 07:09:57
Pengguna Google	2	lama lama harga mahal bgt, pulsa tbtb berkurang,	2026-05-15 02:37:06

		pokonya aneh deh 🙄🙄🙄	
Pengguna Google	5	sangat membantu sukses Slalu telkomsel	2026-05-06 12:15:16

Table 1 shows an example of review data obtained from the scraping process and used as the initial research data.

C. Data Preprocessing

Data preprocessing was carried out to clean and prepare the text data so that it could be processed in the feature extraction and classification stages. This process was designed to address the highly informal and inconsistent writing style typically found in Google Play Store reviews, which often contain slang expressions, non-standard abbreviations, typographical errors, emojis, and mixed use of formal and colloquial Indonesian. The preprocessing process was applied through several stages, starting from data cleaning, case folding, text normalization, tokenizing, stopword removal, and finally stemming, each applied sequentially so that the resulting text representation would be sufficiently standardized for reliable feature extraction.

1) *Data Cleaning*: At this stage, the text is cleaned by selecting and removing elements that are irrelevant to sentiment analysis, including symbols, punctuation marks, emojis, numeric digits, URLs, and other special characters [14].

TABEL II
EXAMPLE OF DATA CLEANING RESULTS

Before Cleaning	After Cleaning
jaringan sangat lelet, harga paket mahal2	jaringan sangat lelet harga paket mahal
lama lama harga mahal bgt, pulsa tbtb berkurang, pokonya aneh deh 🙄🙄🙄	lama lama harga mahal bgt pulsa tbtb berkurang pokonya aneh deh
sangat membantu sukses Slalu telkomsel	sangat membantu sukses Slalu telkomsel

2) *Case Folding*: This stage involves converting the text into lowercase format. This adjustment serves to equate the representation of words with similar meanings that have variations in capitalization, so the system can recognize and process them as a single identical word [15].

TABEL III
EXAMPLE OF FOLDING CASE RESULTS

Before Case Folding	After Case Folding
jaringan sangat lelet harga paket mahal	jaringan sangat lelet harga paket mahal
lama lama harga mahal bgt pulsa tbtb berkurang pokonya aneh deh	lama lama harga mahal bgt pulsa tbtb berkurang pokonya aneh deh
sangat membantu sukses Slalu telkomsel	sangat membantu sukses slalu telkomsel

3) *Text Normalization*: The text normalization stage is carried out to convert non-standard words, abbreviations, slang, and typographical errors into standard word forms

according to the Kamus Besar Bahasa Indonesia (KBBI / Indonesian standard dictionary) or a compiled normalization dictionary [16]. This process aims to reduce variation in word spelling so that words with the same meaning can be represented in a uniform form. The normalization dictionary used in this study was manually compiled based on informal language patterns observed in the dataset, comprising 294 word pairs covering negation forms, common abbreviations, typographical variants, application-specific terms (e.g., "app" to "aplikasi", "login" to "masuk"), and mixed English-Indonesian expressions.

TABEL IV
EXAMPLE OF NORMALIZATION RESULTS

Before Normalization	After Normalization
jaringan sangat lelet harga paket mahal	jaringan sangat lambat harga paket mahal
lama lama harga mahal bgt pulsa tbtb berkurang pokonya aneh deh	lama lama harga mahal banget pulsa tiba tiba berkurang pokoknya aneh
sangat membantu sukses slalu telkomsel	sangat membantu sukses selalu telkomsel

4) *Tokenizing*: The tokenizing stage is carried out by splitting the normalized text into a collection of words, or tokens [17]. Each sentence is separated based on spaces, producing a list of words that can be further processed in the stopword removal and stemming stages. This process was implemented using the word_tokenize function from the NLTK library.

TABEL V
EXAMPLE OF TOKENIZING RESULTS

Before Tokenizing	After Tokenizing
jaringan sangat lambat harga paket mahal	[jaringan, sangat, lambat, harga, paket, mahal]
lama lama harga mahal banget pulsa tiba tiba berkurang pokoknya aneh	[lama, lama, harga, mahal, banget, pulsa, tiba, tiba, berkurang, pokoknya, aneh]
sangat membantu sukses selalu telkomsel	[sangat, membantu, sukses, selalu, telkomsel]

5) *Stopword Removal*: The stopword removal stage is carried out to remove words considered to carry no important information for the sentiment classification process [18]. Words such as conjunctions, prepositions, and other common words are removed because they do not make a significant contribution to determining the sentiment of a review. This process aims to reduce the number of irrelevant words so that only words carrying important information are retained for classification. The stopword list used was based on the NLTK Indonesian stopword corpus, with 23 sentiment-bearing words excluded from removal (e.g., baik, buruk, bagus, jelek, kecewa, mantap, lambat, cepat, mahal, murah, parah, memuaskan, luar, biasa, sangat, recommended, direkomendasikan, tidak, bukan, belum, kurang, tanpa, jangan) because they are critical to sentiment meaning, particularly negation.

TABEL VI
EXAMPLE OF STOPWORD REMOVAL RESULTS

Before Stopword Removal	After Stopword Removal
[jaringan, sangat, lambat, harga, paket, mahal]	[jaringan, sangat, lambat, harga, paket, mahal]
[lama, lama, harga, mahal, banget, pulsa, tiba, tiba, berkurang, pokoknya, aneh]	[harga, mahal, banget, pulsa, berkurang, pokoknya, aneh]
[sangat, membantu, sukses, selalu, telkomsel]	[sangat, membantu, sukses, telkomsel]

6) *Stemming*: The stemming stage is carried out to convert each word into its base form by removing affixes, including prefixes, suffixes, and infixes. This stage aims to equate various word variations that share the same meaning, allowing these variations to be reduced into a single, consistent base form prior to feature extraction. In this study, the stemming process was implemented using the Sastrawi library, which is an implementation of the Nazief-Adriani algorithm for Indonesian text [19].

TABEL VII
EXAMPLE OF STEMMING RESULTS

Before Stemming	After Stemming
[jaringan, sangat, lambat, harga, paket, mahal]	[jaring, sangat, lambat, harga, paket, mahal]
[harga, mahal, banget, pulsa, berkurang, pokoknya, aneh]	[harga, mahal, banget, pulsa, kurang, pokok, aneh]
[sangat, membantu, sukses, telkomsel]	[sangat, bantu, sukses, telkomsel]

D. Sentiment Labeling

After all preprocessing stages were completed, each review was assigned a sentiment label based on the rating value given by the user on the Google Play Store, following a rating-based labeling scheme commonly applied in similar prior studies [2], [3]. Negative sentiment labels were applied to data with rating values of 1 and 2, while positive sentiment labels were assigned to data with rating values of 4 and 5. Meanwhile, reviews with a rating of 3 were excluded from the study because they were considered to represent neutral sentiment, which has the potential to cause ambiguity in the classification process.

To improve labeling quality, label correction was performed on high-rated reviews containing complaint-toned words, using a rule based on comparing the number of negative-toned and positive-toned words in each review; if the frequency of negative words exceeded that of positive words, the sentiment class of the review was corrected to negative. This correction was applied specifically to reviews with a rating of 4 or higher that contained at least one word from a manually compiled negative-toned lexicon of 14 words (mahal, lemot, jelek, buruk, error, gangguan, kecewa, parah, lambat, susah, tidak bisa, ngelag, lag, gagal), matched using word-boundary pattern matching to avoid partial-word matches (e.g., preventing "lag" from matching within "lagi"). For each candidate review, the number of occurrences of words from this negative lexicon was

compared against the number of occurrences of words from a complementary positive-toned lexicon of 21 words (bagus, mantap, puas, memuaskan, baik, suka, recommended, direkomendasikan, oke, lancar, stabil, mudah, praktis, simpel, cepat, murah, keren, terbaik, membantu, nyaman, gampang), following a strict-majority rule rather than a fixed numerical threshold.

Applying this procedure identified 81 candidate reviews, of which 63 were ultimately corrected from positive to negative. The labeling results produced 2,836 reviews, consisting of 1,732 negative sentiments and 1,104 positive sentiments after a number of positive labels were corrected to negative. An example of sentiment labeling results based on rating is shown in Table 8.

TABEL VIII
EXAMPLE OF SENTIMENT LABELING RESULTS BASED ON RATINGS

Rating	Review	Label
1	jaringan sangat lambat harga paket mahal	Negative
2	lama lama harga mahal banget pulsa tiba tiba berkurang pokoknya aneh	Negative
5	sangat bantu sukses selalu telkomsel	Positive

The labeled dataset was subsequently divided into training and testing subsets using an 80:20 split ratio with stratified sampling based on the sentiment label and a fixed random seed (random_state = 42) to ensure reproducibility. This single split was used consistently to derive both the Baseline (Unigram) and N-Gram feature representations, ensuring that the performance comparison between the two scenarios was based on identical training and testing samples.

As part of the data quality check, a duplicate check was also conducted on the labeled dataset (2,836 reviews). No fully duplicate rows were found based on the combination of username, rating, review text, and date; however, 488 reviews had identical original text from different users, increasing to 664 reviews after the text underwent stemming, as the cleaning and normalization processes caused previously distinct texts to converge. These reviews were retained in the dataset, as each represents an independent opinion from a different user.

E. Feature Extraction

Text data that has undergone the preprocessing stage is converted into a numerical representation in the feature extraction process, so it can be further processed by the classification model.

1) **TF-IDF:** As a word-weighting method, Term Frequency-Inverse Document Frequency (TF-IDF) plays a role in transforming text into numerical data [10]. In this study, TF-IDF was applied in two scenarios, namely Baseline using unigram (ngram_range = (1,1)) and N-Gram using a combination of unigram and bigram (ngram_range = (1,2)), as the feature representation used by the Naive Bayes and Support Vector Machine (SVM) algorithms.

$$TFIDF(t, d) = TF(t, d) \times IDF(t)$$

The TF-IDF value is obtained by multiplying Term Frequency (TF) by Inverse Document Frequency (IDF). TF represents the frequency of occurrence of a term in a particular document, while IDF measures the relevance or importance of that term based on its distribution across the entire document collection. The mathematical formulas for TF and IDF are shown below.

$$TF(t, d) = \frac{f_{t,d}}{\sum_k f_{k,d}}$$

In this equation, $(f_{t,d})$ indicates the frequency of occurrence of term t in document d , while $(\sum_k f_{k,d})$ represents the total number of terms contained in that document.

$$IDF(t) = \log\left(\frac{N}{df(t)}\right)$$

The variable (N) is defined as the total number of documents, while $(df(t))$ measures the number of documents containing term t . The feature matrix produced from the TF-IDF calculation is then used as input for the Naive Bayes and Support Vector Machine algorithms in the classification testing.

Both feature representations were fit exclusively on the training data to prevent data leakage, with the fitted vectorizer subsequently applied to transform the testing data using the transform operation only. The resulting feature dimensions differed between the two scenarios: the Baseline (Unigram) representation produced 2,726 features without a minimum document frequency filter, resulting in training and testing matrices of size $2,268 \times 2,726$ and $568 \times 2,726$, respectively, while the N-Gram (Unigram + Bigram) representation, filtered using a minimum document frequency of 2 to control vocabulary growth, produced 2,295 features, resulting in matrices of size $2,268 \times 2,295$ and $568 \times 2,295$, respectively. It should be noted that the N-Gram representation contains fewer features than the Baseline despite incorporating bigrams; this is because the minimum document frequency filter was applied only to the N-Gram scenario and not to the Baseline, so the difference in vocabulary size reflects the filtering configuration rather than an inherent reduction in feature richness caused by including bigrams.

2) **N-Gram:** N-Gram is a feature extraction technique that forms sequences of words based on a number of adjacent tokens within a document. This technique is used to preserve relationships between words so that it can capture sentence context that a single token cannot represent [10]. The formation of an N-Gram is expressed by the following equation.

$$G_n = (w_1, w_2, \dots, w_n)$$

Where G_n denotes an N-Gram consisting of (n) consecutive tokens, while $(w_1, w_2, \dots, w_n)$ are the tokens

that make up the N-Gram. In this study, a combination of unigram and bigram ($ngram_range = (1,2)$) was used so the model could represent both single words and pairs of words occurring consecutively. The resulting N-Gram is then weighted using TF-IDF, producing a feature representation that better preserves context between words in user reviews.

F. Classification Models

1) *Naive Bayes*: Naive Bayes utilizes Bayes' Theorem to perform probabilistic classification by assuming independence among all features [3]. This method is ideal for text classification due to its lightweight computation and its ability to handle high-dimensional TF-IDF feature representations. The probability of a document belonging to a class is calculated using the following equation.

$$P(C | X) = \frac{P(X | C) \times P(C)}{P(X)}$$

Where $P(C | X)$ describes the posterior probability value of a document with respect to class C . Meanwhile, $P(X | C)$ is the probability of feature X given class C , $P(C)$ refers to the prior probability of the class, and $P(X)$ indicates the probability distribution of the feature across the entire dataset. Unlike SVM, the Multinomial Naive Bayes implementation used in this study does not require a random seed, as its parameter estimation is fully deterministic given a fixed training set.

2) *Support Vector Machine*: As a classification method, Support Vector Machine (SVM) works by finding the best hyperplane that maximizes the separating margin between classes [10]. This algorithm is a primary choice in text analysis due to its effectiveness and performance in high-dimensional data spaces. This study utilizes a linear kernel in SVM to classify features resulting from TF-IDF weighting, with the mathematical function formulated as follows.

$$f(x) = w^T x + b$$

In this equation, w is defined as the weight vector, x represents the feature vector, and b is the bias value.

3) *Grid Search*: Grid Search is a hyperparameter validation method used to systematically evaluate various parameter combinations in order to obtain the most suitable model configuration [13]. This study applies the Grid Search technique combined with 5-fold cross validation on the Support Vector Machine (SVM) algorithm to optimize the hyperparameters C , γ , and kernel type. The parameter grid tested consisted of C values of {0.1, 1, 10, 100}, γ values of {scale, auto, 0.1, 0.01}, and kernel types of {linear, RBF}, resulting in 32 parameter combinations evaluated for each feature representation scenario, with model selection based on the mean F1-score macro average across the 5 cross-validation folds. A fixed random seed ($random_state = 42$) was applied consistently across all SVM experiments, including both the baseline

models and the Grid Search validation, to ensure reproducibility of the results. Grid Search validation was applied specifically to SVM because this algorithm achieved higher accuracy than Naive Bayes in both feature representation scenarios prior to hyperparameter tuning (92.43% versus 88.91% for the Baseline scenario, and 91.90% versus 89.26% for the N-Gram scenario, as shown in Table IX and Table X). This approach refers to a hyperparameter tuning strategy specifically proven effective in improving SVM classification accuracy [12]. This validation was performed separately for both feature representation scenarios, namely Baseline (Unigram) and N-Gram, using an identical evaluation scheme to ensure that the performance comparison between the two scenarios was conducted fairly. The selection of the best parameter combination for each scenario was based on the F1-score macro average value from cross validation.

G. Evaluation Metrics

The evaluation process aims to assess and compare model performance in classifying sentiment across each research scenario, namely Baseline, N-Gram, and the SVM model that had undergone hyperparameter validation using Grid Search. The evaluation metrics used include accuracy, precision, recall, F1-score, and confusion matrix. The use of accuracy or F-measure metrics alone must be done with caution, as they have the potential to provide inaccurate evaluations on datasets with imbalanced class distributions [20], so several evaluation metrics are used simultaneously in this study to present a more thorough and in-depth picture of model performance.

1) *Accuracy*: The accuracy metric is used to assess the extent to which the model is able to correctly predict test data. The accuracy formula is shown in the equation below.

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

Where the variable TP (True Positive) and TN (True Negative) each represent correct prediction results, while FP (False Positive), and FN (False Negative) represent model prediction errors.

2) *Precision*: The precision metric is used to assess the proportion of correct prediction results with respect to the predicted sentiment class. The precision formula is shown below.

$$Precision = \frac{TP}{TP + FP}$$

3) *Recall*: The recall metric measures the extent of the model's ability to capture all actual data belonging to a particular sentiment class. The recall value is calculated based on the equation below.

$$Recall = \frac{TP}{TP + FN}$$

4) *F1-Score*: The trade-off value between precision and recall is measured through the F1-Score, which provides a fairer performance evaluation of the model for each sentiment category. The F1-score formula is shown as follows.

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

5) *F1-Score Macro Average*: Balanced model performance evaluation for both the positive and negative classes is measured using the F1-score macro average (F1-Macro). This metric is important under conditions of imbalanced class distribution, because it does not give greater weight to the class with a larger amount of data [21], [22].

$$F1_{\text{Macro}} = \frac{F1_{\text{Negatif}} + F1_{\text{Positif}}}{2}$$

6) *Confusion Matrix*: The classification precision details for each sentiment class are presented through a confusion matrix based on a compilation of TP, TN, FP, and FN. In addition to helping detect patterns of modeling errors, this matrix is also used as the main reference in comparing performance between models [23].

III. RESULTS AND DISCUSSION

A. Feature Extraction Results

The feature extraction stage utilizes the TF-IDF technique to transform preprocessed text data into a numerical matrix ready to be processed by the classification algorithm. Feature extraction was carried out in two scenarios, namely Baseline using unigram (ngram_range = (1,1)) and TF-IDF with N-Gram features using a combination of unigram and bigram (ngram_range = (1,2)).

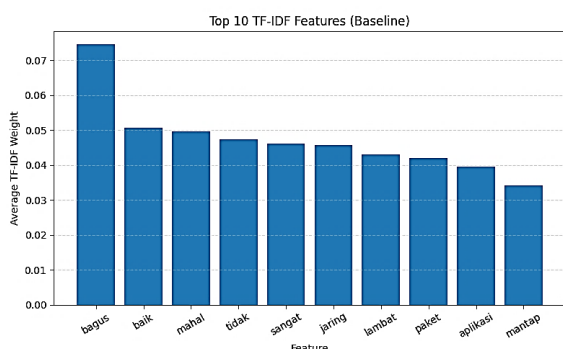


Figure 2 Top 10 Weighted TF-IDF Features (Baseline)

Based on Figure 2, the words bagus ("good"), baik ("good/nice"), mahal ("expensive"), and tidak ("not/no") have the highest TF-IDF weights in the Baseline scenario. The appearance of the word "tidak" ("not") as one of the high-weighted features indicates that negation words were successfully preserved during preprocessing, allowing them to be recognized as significant features in the TF-IDF

representation. These features are subsequently used as input for the classification process.

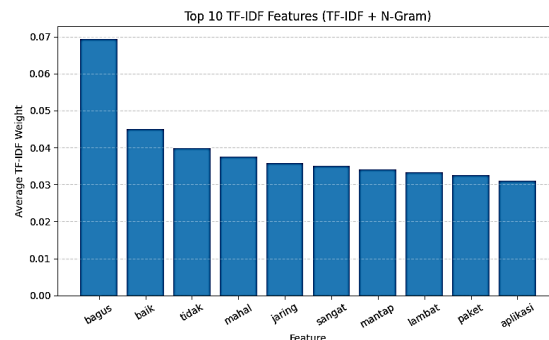


Figure 3 Top 10 Weighted TF-IDF Features (N-Gram)

Based on Figure 3, the features with the highest TF-IDF weights are still dominated by the words bagus ("good"), baik ("good/nice"), tidak ("not/no"), and mahal ("expensive") after applying the unigram and bigram combination. The similar pattern of feature dominance compared to the Baseline scenario indicates that the N-Gram representation does not significantly change the main features contributing to sentiment classification.

B. Classification Performance

The comparison of model effectiveness and performance in each scenario is measured based on accuracy, precision, recall, and F1-score metrics. This evaluation stage is intended to identify the extent to which the feature representation technique affects the classification results for both the Support Vector Machine (SVM) and Naive Bayes algorithms.

1) *Baseline Models*: This scenario is used as an initial benchmark to measure the performance of the Naive Bayes and Support Vector Machine (SVM) algorithms by applying unigram-based TF-IDF feature representation.

TABEL IX
PERFORMANCE EVALUATION OF BASELINE MODELS

Metric	Naive Bayes	SVM
Accuracy	88.91%	92.43%
Precision	95.40%	91.98%
Recall	75.11%	88.24%
F1-Score	84.05%	90.07%
F1-Score (Macro)	87.77%	91.98%

The baseline scenario experiment results show that the performance achieved by Support Vector Machine (SVM) exceeds that obtained by the Naive Bayes algorithm. SVM obtained an accuracy of 92.43% with an F1-score of 90.07%, while Naive Bayes obtained an accuracy of 88.91% with an F1-score of 84.05%. Although Naive Bayes has a high precision value (95.40%), its lower recall value (75.11%) indicates that the model still misses a portion of the positive sentiment data. This performance difference is

also reflected in the F1-score macro average, where SVM obtained 91.98% compared to Naive Bayes' 87.77%, showing that SVM remains superior in a balanced manner across both sentiment classes, not only on the majority class.

2) *N-Gram Models*: This scenario uses TF-IDF with a combination of unigram and bigram to evaluate the effect of adding N-Gram on the performance of the Support Vector Machine (SVM) and Naive Bayes algorithms.

TABEL X
PERFORMANCE EVALUATION OF TF-IDF WITH N-GRAM MODELS

Metric	Naive Bayes	SVM
Accuracy	89.26%	91.90%
Precision	94.44%	91.47%
Recall	76.92%	87.33%
F1-Score	84.79%	89.35%
F1-Score (Macro)	88.24%	91.41%

The results show that the addition of N-Gram features improves performance for Naive Bayes, with accuracy increasing from 88.91% to 89.26% and F1-score increasing from 84.05% to 84.79%. However, for the SVM algorithm, the addition of N-Gram instead results in slightly lower accuracy compared to the Baseline scenario, namely from 92.43% to 91.90%, with the F1-score decreasing from 90.07% to 89.35%. These results indicate that the effect of N-Gram representation on classification performance depends on the algorithm used, and is not always directly proportional to a performance improvement.

3) *Comparison Across Algorithms*: The accuracy comparison across algorithms for all tested feature representation scenarios is shown in Figure 4 below.

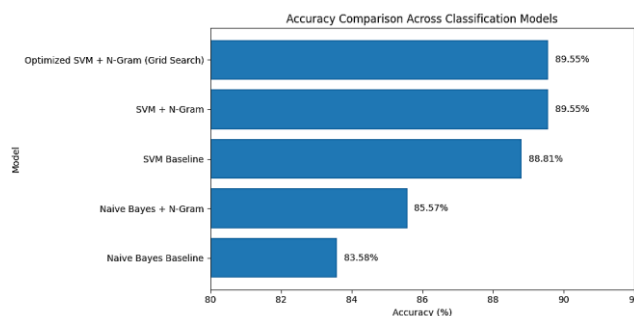


Figure 4 Accuracy Comparison Across Models

Figure 4 shows the accuracy comparison across algorithms for all tested feature representation scenarios. SVM Baseline obtained the highest accuracy at 92.43%, followed by SVM + N-Gram at 91.90%, while Naive Bayes Baseline produced the lowest accuracy at 88.91%. These results show that, within the scope of this study, the SVM algorithm outperforms Naive Bayes across both feature representation scenarios, while the effect of N-Gram on SVM performance actually results in a slight decrease compared to the Unigram representation.

C. Hyperparameter Validation

To ensure that the model configuration used was appropriate, the best hyperparameter validation was performed for the Support Vector Machine (SVM) model using Grid Search with 5-fold cross validation. This validation was applied separately to both feature representation scenarios, namely Baseline (Unigram) and N-Gram. The parameters validated included kernel, C, and gamma, while the selection of the best parameter combination for each scenario was based on the F1-score macro average value from cross validation.

1) *Grid Search Result*: The Grid Search validation results for both feature representation scenarios are shown in Table XI below.

TABEL XI
BEST HYPERPARAMETER VALIDATION RESULTS

Parameter	Best Value
Kernel	Linear
C	1
Gamma	Scale
5-Fold CV F1-Macro (Baseline/Unigram)	92.60%
5-Fold CV F1-Macro (N-Gram)	92.23%

Based on Table XI, the Grid Search validation results for both scenarios produced the same hyperparameter combination, namely kernel = linear, C = 1, and gamma = scale. The Baseline (Unigram) scenario obtained a 5-fold cross validation F1-Macro of 92.60%, slightly higher than the N-Gram scenario at 92.23%. The parameter combination found is identical to the configuration already used in the previous modeling stage, so this validation result confirms that the configuration used from the beginning was already appropriate for both feature representations. Based on the comparison of the F1-score macro average on the testing data, SVM Baseline (Unigram) validated with Grid Search obtained an F1-Macro of 91.98%, slightly higher than SVM + N-Gram validated with Grid Search at 91.41%. Therefore, the final model established in this study is SVM Baseline (Unigram) with Grid Search.

Since the linear kernel proved to be optimal, the gamma parameter had no actual effect on the final model, as it only influences non-linear kernels such as RBF, polynomial, or sigmoid and was included in the Grid Search grid merely for testing completeness. Accordingly, the parameters that effectively determine the final model's performance are limited to the kernel type and the regularization parameter C.

2) *Comparison After Grid Search Validation*: The accuracy comparison between the two scenarios after the Grid Search validation process is shown in Figure 5.

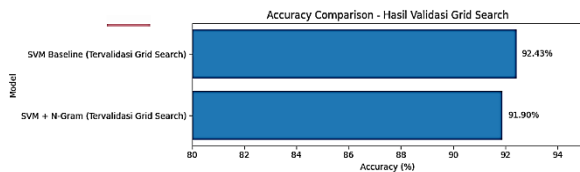


Figure 5 Comparison of Grid Search Validation Results

Figure 5 shows the accuracy comparison between the two scenarios after hyperparameter validation using Grid Search. SVM Baseline (Unigram) remains superior with an accuracy of 92.43%, compared to SVM + N-Gram at 91.90%. This result is consistent with the comparison before Grid Search validation, showing that the validation process does not change the conclusion that Unigram representation provides better performance in this study.

D. Confusion Matrix of the Final Model

Based on the comparison results in the previous section, the final model established for this study is SVM Baseline (Unigram) with Grid Search. To determine the final model's ability to classify each sentiment class in more detail, an analysis was carried out using a Confusion Matrix on the testing data.

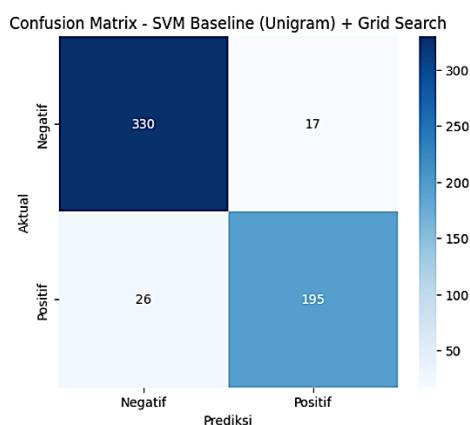


Figure 6 Confusion Matrix of the Final Model

Based on Figure 6, the final model successfully classified 330 negative sentiment data and 195 positive sentiment data correctly, while prediction errors occurred for 17 negative data points misidentified as positive, and 26 positive samples detected as negative. The classification error rate for the negative class is relatively low (17 of 347 data points, approximately 4.9%), as is the case for the positive class (26 of 221 data points, approximately 11.8%). This result is consistent with the precision (91.98%) and recall (88.24%) values for the positive class presented previously, and shows that the final model is able to recognize both sentiment classes well, with a slight tendency to be better at recognizing negative sentiment than positive sentiment.

In terms of error type, the model produced more false negatives (26 positive reviews misclassified as negative) than false positives (17 negative reviews misclassified as

positive). This indicates that the model has a slightly stronger tendency to under-predict positive sentiment rather than to over-predict it, which is consistent with the lower recall (88.24%) compared to precision (91.98%) observed for the positive class. To further account for the class distribution in the testing data, the balanced accuracy of the final model was also calculated from the confusion matrix, yielding a value of 91.67% (the average of 88.24% recall for the positive class and 95.10% specificity, i.e., 330/347, for the negative class). This value is close to the overall accuracy of 92.43%, indicating that the reported accuracy is not substantially inflated by the class distribution in the testing data.

Taken together, these evaluation results indicate that the final model performs reliably across both sentiment classes. When compared to prior studies on MyTelkomsel and similar telecommunication application reviews, this accuracy is also competitive, exceeding the 83% accuracy reported using an expert-labeled dataset [1], and closely approaching, though slightly below, the 93.36% accuracy reported in a related MyTelkomsel study using SVM and Logistic Regression [2]. Beyond raw accuracy, the present study further contributes a systematic comparison of Unigram and N-Gram feature representations and a fair, scenario-separated hyperparameter validation that were not addressed in these prior studies.

E. Sentiment Visualization

Sentiment visualization is used to obtain a general picture of the words that most frequently appear in each sentiment class, performed using WordCloud on the review data that had undergone the preprocessing stage.

1) Positive Wordcloud



Figure 7 Positive Sentiment Word Cloud

Based on Figure 7, the words that most represent positive sentiment include sangat ("very"), baik ("good"), bagus ("good"), telkomsel ("Telkomsel"), tidak ("not/no"), aplikasi ("application"), mantap ("great"), bantu ("help"), and mudah ("easy"). The dominance of these words shows that users gave positive assessments of the application's quality, service, and ease of use.

study served more as a confirmation of the already-optimal initial configuration than as a means of improving performance.

It should be noted that Grid Search was applied only to SVM, as it already outperformed Naive Bayes in both feature representation scenarios prior to tuning (92.43% versus 88.91% for Baseline, and 91.90% versus 89.26% for N-Gram), and Naive Bayes offers a considerably narrower hyperparameter space. This asymmetry in tuning scope represents a limitation of the current comparative analysis, and future work could explore tuning its alpha parameter for a more complete comparison. In addition, the reported superiority of SVM is based on a descriptive comparison of evaluation metrics rather than a formal statistical significance test (e.g., McNemar's test), which represents a limitation to be addressed in future research.

Overall, this study shows that TF-IDF Unigram (Baseline) performs equal to or better than Unigram+Bigram on this dataset, particularly for SVM, likely because negation words were preserved during preprocessing, allowing single-word features to capture sentiment effectively, while added bigram features introduce sparsity and dilute informative unigram terms in informal Indonesian text. The results also confirm that SVM is more effective than Naive Bayes for this task, offering a reference for future research through larger datasets or more diverse feature extraction methods and algorithms.

REFERENCES

- [1] A. I. A. Azkiya and B. Santoso, "Penerapan Machine Learning Pada Analisis Sentimen Aplikasi MyTelkomsel Menggunakan Data Ulasan Google Playstore," *Indones. J. Comput. Sci.*, vol. 13, no. 1, pp. 1227–1240, 2023, [Online]. Available: <http://ijcs.stmikindonesia.ac.id/ijcs/index.php/ijcs/article/view/3135>
- [2] R. Bagus Adinata and D. Laily Fithri, "Sentiment Classification of MyTelkomsel Reviews Using SVM and Logistic Regression," *IJCCS (Indonesian J. Comput. Cybern. Syst.)*, vol. 20, no. 1, pp. 83–94, 2026, doi: 10.22146/ijccs.110409.
- [3] T. S. Ningsih, T. I. Hermanto, and I. M. Nugroho, "Sentiment Analysis of Mobile Provider Application Reviews Using Naive Bayes Algorithm and Support Vector Machine," *Sink. J. dan Penelit. Tek. Inform.*, vol. 8, no. 2, pp. 824–835, 2024, doi: <https://doi.org/10.33395/v8i2.13469>.
- [4] A. Puji Astuti, S. Alam, and I. Jaelani, "Komparasi Algoritma Support Vector Machine dengan Naive Bayes Untuk Analisis Sentimen Pada Aplikasi BRImo," *J. Bangkit Indones.*, vol. 11, no. 2, pp. 1–6, 2022, doi: 10.52771/bangkitindonesia.v11i2.196.
- [5] A. Nurian, "Analisis Sentimen Ulasan Pengguna Aplikasi Google Play Menggunakan Naive Bayes," *J. Inform. dan Tek. Elektro Terap.*, vol. 11, no. 3s1, pp. 829–835, 2023, doi: 10.23960/jitet.v11i3s1.3348.
- [6] J. Homepage et al., "Analisis Sentimen Pengguna Aplikasi Jamsostek Mobile Berdasarkan Ulasan Google Play Store Menggunakan Algoritma Support Vector Machine dan Naive Bayes," *MALCOM Indones. J. Mach. Learn. Comput. Sci.*, vol. 6, no. 1, pp. 373–384, 2026, [Online]. Available: <https://journal.irpi.or.id/index.php/malcom/article/view/2526>
- [7] M. Zainuddin Siregar, A. Marwan Elhanafi, and D. Irwan, "Analisis Sentimen Terhadap Ulasan Aplikasi Media Sosial Di Google Play Menggunakan Algoritma Naive Bayes," *JATI (Jurnal Mhs. Tek. Inform.)*, vol. 9, no. 2, pp. 3373–3381, 2025, doi: 10.36040/jati.v9i2.12841.
- [8] J. M. Manampiring, J. Halawa, J. Zai, I. Kurniawati, R. Fahlap, and W. Bismi, "Analisis Sentimen Ulasan Aplikasi X Pada Google Play Store Menggunakan Komparasi Svm, Knn, Dan Naive Bayes," *Djtechno J. Teknol. Inf.*, vol. 7, no. 1, pp. 1–13, 2026, doi: 10.46576/djtechno.v7i1.8095.
- [9] F. Marzuqi, Y. Purba, T. Syaputra, W. Bismi, I. Kurniawati, and R. Fahlap, "Perbandingan Algoritma Machine Learning Untuk Sentimen Ulasan Aplikasi Jobstreet," *INFOTECH J.*, vol. 11, no. 2, pp. 416–423, 2025, doi: 10.31949/infotech.v11i2.16609.
- [10] F. Amin Mujtahid, B. Zaman, and G. Guntoro Aji, "Analisis Klasifikasi Sentimen Neobank: Perbandingan Konfigurasi N-Gram pada TF-IDF Menggunakan Naive Bayes dan SVM," *TIN Terap. Inform. Nusan.*, vol. 6, no. 12, pp. 2375–2383, 2026, doi: 10.47065/tin.v6i12.9702.
- [11] F. Lawalata, "Analisis Performa Support Vector Machine (SVM) dengan Hyperparameter Tuning untuk Klasifikasi Data Stress Monitoring Siswa," *JOINTER J. Informatics Eng.*, vol. 6, no. 02, pp. 9–15, 2025, doi: 10.53682/jointer.v6i02.413.
- [12] E. Elgeldawi, A. Sayed, A. R. Galal, and A. M. Zaki, "Hyperparameter tuning for machine learning algorithms used for arabic sentiment analysis," *Informatics*, vol. 8, no. 4, pp. 1–21, 2021, doi: 10.3390/informatics8040079.
- [13] W. Nugraha and A. Sasongko, "Hyperparameter Tuning on Classification Algorithm with Grid Search," *Sistemasi*, vol. 11, no. 2, p. 391, 2022, doi: 10.32520/stmsi.v11i2.1750.
- [14] O. Alotaibi, E. Pardede, and S. Tomy, "Cleaning Big Data Streams: A Systematic Literature Review," *Technologies*, vol. 11, no. 4, pp. 1–24, 2023, doi: 10.3390/technologies11040101.
- [15] K. P. Harmandini and K. M. L., "Analysis of TF-IDF and TF-RF Feature Extraction on Product Review Sentiment," *Sinkron*, vol. 8, no. 2, pp. 929–937, 2024, doi: 10.33395/sinkron.v8i2.13376.
- [16] M. A. Palomino and F. Aider, "Evaluating the Effectiveness of Text Pre-Processing in Sentiment Analysis," *Appl. Sci.*, vol. 12, no. 17, 2022, doi: 10.3390/app12178765.
- [17] A. Gasparetto, M. Marcuzzo, A. Zangari, and A. Albarelli, "A Survey on Text Classification Algorithms: From Text to Predictions," *Inf.*, vol. 13, no. 2, pp. 1–39, 2022, doi: 10.3390/info13020083.
- [18] A. B. Putra Negara, "The Influence Of Applying Stopword Removal And Smote On Indonesian Sentiment Classification," *Lontar Komput. J. Ilm. Teknol. Inf.*, vol. 14, no. 3, p. 172, 2023, doi: 10.24843/lkjiti.2023.v14.i03.p05.
- [19] J. Jumadi, D. S. Maylawati, L. D. Pratiwi, and M. A. Ramdhani, "Comparison of Nazief-Adriani and Paice-Husk algorithm for Indonesian text stemming process," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1098, no. 3, p. 032044, 2021, doi: 10.1088/1757-899x/1098/3/032044.
- [20] L. Lavazza and S. Morasca, "Common Problems With the Usage of F-Measure and Accuracy Metrics in Medical Research," *IEEE Access*, vol. 11, no. June, pp. 51515–51526, 2023, doi: 10.1109/ACCESS.2023.3278996.
- [21] K. M. Sujon, R. Hassan, K. Choi, and M. A. Samad, "Accuracy, precision, recall, F1-score, or MCC Empirical evidence from advanced statistics, ML, and XAI for evaluating business predictive models," *J. Big Data*, vol. 12, 2025, doi: 10.1186/s40537-025-01313-4.
- [22] M. C. Hinojosa Lee, J. Braet, and J. Springael, "Performance Metrics for Multilabel Emotion Classification: Comparing Micro, Macro, and Weighted F1-Scores," *Appl. Sci.*, vol. 14, no. 21, 2024, doi: 10.3390/app14219863.
- [23] I. Markoulidakis and G. Markoulidakis, "Probabilistic Confusion Matrix: A Novel Method for Machine Learning Algorithm Generalized Performance Analysis," *Technologies*, vol. 12, no. 7, 2024, doi: 10.3390/technologies12070113.