Jurnal Esensi Infokom Vol 10 No. 2 Oktober 2026
e-ISSN: 2828-6707
https://esensijournal.com/index.php/infokom
DOI: 10.55886/infokom.v10i2.545

Comparison of Sentiment Classification Algorithms User Reviews of Qasir App on Google Play Store

Essy Malays Sari Sakti[1*], Ahmad Soderi[2], Minda Mora Purba[3], Asnefi[4]

[1,]*Informatics Study Program, YAI. Persada Indonesia University*
E-Mail P. Diponegoro No.74 . Central Jakarta, INDONESIA

[1] emalays67@gmail.com

[2,] *Information Systems Study Program, STMIK Mercusuar*

[2] *ahmad@.mercusuar.ac.id*

[3]*Information Systems Study Program, Marsekal Suryadarma Aerospace University*

Jl. Raya Halim Perdanakusuma, East Jakarta. INDONESIA

[3] *minda@unsurya.ac.id*

[4]*Informatics Engineering Study Program, Patria Artha University Jl.Tun Abdul Razak, Paccinongga, South Sulawesi, INDONESIA*

[4] asnefi@patria-artha.ac.id

***Abstract*— The Qasir application is one of the Point of Sale (POS) platforms widely used by Micro, Small, and Medium Enterprises (MSMEs) in Indonesia to manage transactions and business operations. User reviews on the Google Play Store platform provide valuable input on system performance, feature reliability, and user satisfaction levels. This study aims to compare the performance of three machine learning classification algorithms, namely Naive Bayes, Support Vector Machine (SVM), and Random Forest, in classifying user review sentiment into positive and negative categories. The dataset consists of 2,500 reviews obtained through web scraping techniques from the Google Play Store. The pre-processing stage of data includes text cleaning, case folding, tokenization, stopword removal, stemming, and feature extraction using Term Frequency-Inverse Document Frequency (TF-IDF). The model evaluation was carried out using a 10-fold cross-validation technique based on the Accuracy, Precision, Recall, and F1-Score metrics. The results of the experiment showed that the Support Vector Machine (SVM) algorithm produced the highest performance with an accuracy of 85.20%, Precision of 84.80%, Recall of 85.10%, F1-Score of 84.95%, and a standard deviation of 1.25%, outperforming Random Forest (82.40% accuracy) and Naïve Bayes (76.60% accuracy). These results prove that TF-IDF-based SVM is very effective in the sentiment classification of Indonesian texts and can provide strategic recommendations for developers to improve the service quality of the Qasir application**

Keywords— Sentiment Classification, Qasir Application, Support Vector Machine, Naïve Bayes, Random Forest, TF-IDF.

Intisari— **Aplikasi Qasir merupakan salah satu platform Point of Sale (POS) yang banyak digunakan oleh Pelaku Usaha Mikro, Kecil, dan Menengah (UMKM) di Indonesia untuk mengelola transaksi serta operasional bisnis. Ulasan pengguna pada platform Google Play Store memberikan masukan berharga mengenai kinerja sistem, keandalan fitur, dan tingkat kepuasan pengguna. Penelitian ini bertujuan untuk membandingkan performa tiga algoritma klasifikasi machine learning yaiut Naive Bayes, Support Vector Machine (SVM), dan Random Forest, dalam mengklasifikasikan sentimen ulasan pengguna ke dalam kategori positif dan negatif. Dataset terdiri dari 2.500 ulasan yang diperoleh melalui teknik web scraping dari Google Play Store. Tahap pra-pemrosesan data meliputi text cleaning, case folding, tokenization, stopword removal, stemming, serta ekstraksi fitur menggunakan Term Frequency-Inverse Document Frequency (TF-IDF). Evaluasi model dilakukan menggunakan teknik 10-fold cross-validation berdasarkan metrik Akurasi, Presisi, Recall, dan F1-Score. Hasil eksperimen menunjukkan bahwa algoritma Support Vector Machine (SVM) menghasilkan performa tertinggi dengan akurasi sebesar 85,20%, Presisi 84,80%, Recall 85,10%, F1-Score 84,95%, dan standar deviasi 1.25 %, mengungguli Random Forest (akurasi 82,40%) dan Naïve Bayes (akurasi 76,60%). Hasil ini membuktikan bahwa SVM berbasis TF-IDF sangat efektif dalam klasifikasi sentimen teks berbahasa Indonesia dan dapat memberikan rekomendasi strategis bagi pengembang untuk meningkatkan kualitas layanan aplikasi Qasir**

Kata kunci— Klasifikasi Sentimen, Aplikasi Qasir, Support Vector Machine, Naïve Bayes, Random Forest, TF-IDF

.

I. INTRODUCTION

The development of information technology has encouraged the digitalization of the Micro, Small, and Medium Enterprises (MSMEs) sector in Indonesia. One form of adaptation of this technology is the use of mobile-based Point of Sale (POS) applications to make it easier to record transactions, manage stock of goods, and make daily financial reports [1] [2]. The Qasir application is one of the POS solutions that is very helpful for MSME operations, open reviews provided by users on the application distribution platform often reflect various levels of satisfaction and technical constraints. Common problems that

e-ISSN: 2828-6707
https://esensijournal.com/index.php/infokom
DOI: 10.55886/infokom.v10i2.545

are often complained about include data synchronization between devices, receipt printing failures, feature subscription fees, and user interface responsiveness. These textual reviews store valuable information that can be used to measure the quality of application services.

However, manually grouping and analyzing thousands of reviews takes considerable time and resources. Therefore, an automated approach uses sentiment-based analysis *Machine Learning* indispensable [3].

Sentiment analysis is a field in natural language processing (*Natural Language Processing*) that focuses on identifying, extracting, and grouping the opinions or levels of satisfaction contained in the text into specific sentiment categories such as positive and negative [4]. In the mobile app domain, user open reviews on the Google Play Store platform store qualitative data that reflects the user experience (*user experience*) as well as technical system constraints [5]. Automatic textual review processing allows developers to evaluate systems and feature improvements more quickly and in a more structured manner. Classification methods used by Support Vector Machine (SVM), Random Forest, Naive Bayes.

Support Vector Maine (SVM) is a supervised learning algorithm that works on the principle of geometric separation [6]. SVM aims to find the best linear separator plane called a hyperplane to separate the dataset into two different classes, such as positive and negative sentiment classes [7]. SVM is very resilient in handling qualitative data from feature extraction (such as TF-IDF Vector weights) which has a very dense feature space (sparse data) and has proven to be effective in preventing overfitting [8] [9].

Random Forest is an Ensemble Learning-based algorithm that combines prediction results from a set of individual decision trees to predict end-grade labels [10]. The formation of the Random Forest model utilizes Bootstrap Aggregating (Bagging) and Feature Randomness techniques. In the training stage, the algorithm randomly takes a bootstrap sample along with a subset of word features to build hundreds of decision trees that are independent of each other. Each decision tree in the "forest" will independently predict the new input data [11].

Naive Bayes is one of the probabilistic-based classification algorithms based on Bayes' Theorem [12]. This algorithm works by calculating the posterior odds of a class based on a combination of features present in the review text data [13]. The basic assumption of this algorithm lies in a strong conditional independence, where the appearance of a word in a sentence is considered to have no direct relation or influence on the occurrence of other words [14]. In sentiment classification, Naïve Bayes determines the class label (positive or negative) by selecting the highest probability value from the result of multiplying the prior probability and the probability of likelihood of each term.

This study aims to compare the performance of Naive Bayes, Support Vector Machine (SVM), and Random Forest algorithms in classifying the sentiment of user reviews of the Qasir application on the Google Play Store. The results of this comparison are expected to make an academic contribution to the selection of the most optimal text classification algorithm and provide practical input for Qasir application developers in improving services.

II. LITERATURE REVIEW

Much has been done in the development of sentiment analysis on mobile app reviews in Indonesia, but most previous studies have been limited to algorithm variation as well as domain-specific coverage. Previous research has focused on the Support Vector Machine (SVM) algorithm to classify financial services sentiment (Kredivo) [15], while another study evaluated the use of Naive Bayes to map the polarity of user sentiment on the quality of MyXL applications through a text mining approach using the knowledge discovery from data (KDD) methodology with the evaluation of model testing using the Confusion Matrix resulting in a high accuracy rate of 85.28% [16]. Meanwhile, another study reviewed an operational application (MyXL) which produced an accuracy of 61.5% due to class imbalance constraints and the use of non-standard language [17]. On the other hand, previous research compared SVM and Random Forest (RF) to handle the complexity of text in the form of sarcasm detection on YouTube social media. Especially in the domain of Point of Sale (POS) applications for MSMEs such as the Qasir application, have conducted a comparative analysis, but the discussion is still limited to the comparison of two algorithms, namely SVM and Random Forest [18],. Apart from the contributions of these previous studies, there is a research gap that has not been fully met, namely: there has not been a study that directly compares three different machine learning paradigms, namely the field separation method *(hyperplane*) such as SVM, probability methods such as Naïve Bayes, and ensemble tree methods such as Random Forest in an experimental framework in the domain of digital qasir application reviews. The characteristics of user reviews of POS applications such as Qasir have their own unique vocabulary, where reviews are often related to the technical features of business operations combined with non-standard languages typical of Google Play Store users. In the absence of a comprehensive algorithmic comparator, it is difficult to determine which model is the most superior and stable in processing the characteristics of these specific texts. To fill the research gap, this study offers a number of novelties (*novelty)* In the form of:

A. Comprehensive multi-algorithm comparison,

This study did not only test one or two algorithms, but compared three algorithms from three different machine learning paradigms (SVM, Random Forest, and Naïve Bayes) on the same Indonesian review dataset.

B. Complete metric-based performance evaluation.

The test not only uses the Accuracy metric, but also evaluates the Precision, Recall, and F1-Score values to measure the resilience of each algorithm in handling review class imbalances on the Google Play Store

Jurnal Esensi Infokom Vol 10 No. 2 Oktober 2026
e-ISSN: 2828-6707
https://esensijournal.com/index.php/infokom
DOI: 10.55886/infokom.v10i2.545

III. RESEARCH METHODOLOGY

The methodology of this research consists of five main stages: review data collection, text preprocessing, text feature extraction, classification algorithm modeling, and model performance evaluation. These stages are presented in figure 1.

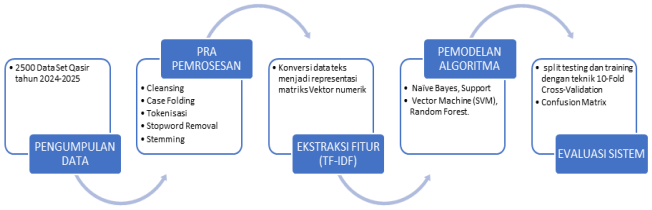


Figure 1. Research Stages

A. Data Collection

User review data is automatically downloaded from the Python programming language Google Play Store. The collected dataset totals 2,500 reviews published in the 2024–2025 range. These reviews were then labeled into three sentiment categories: positive, negative, and neutral to serve as the primary dataset for training and testing the model. The sentiment labeling results are shown in Table I

TABLE I. SUMMARY OF DATASET DISTRIBUTION

No.	Sentiment	Count
1.	Positive	1.425
2.	Negative	825
3.	Neutral	250
Total		2500

B. Text Pre-Processing

Text pre-processing aims to eliminate the noise element in the review so that the text is ready to be processed by machine learning algorithms. The stages of text pre-processing are presented in table II.

TABLE II. STAGES OF PRE-TEXT PROCESSING

Stages	Description	Toos/ Library
Cleaning	Remove punctuation, numbers, symbols, emojis, and URL links.	*Python (re) Regular Expressions*
Case Folding	Converts the entire text to lowercase.	Python .lower()
Tokenization	Breaking sentences into single words (tokens).	NLTK, word_tokenize()
Stopword Removal	Omit common words (e.g.: "which", "in", "and", "to").	Sastrawi. StopWordRemover Factory
Stemming	Convert a root word to a root word	Sastrawi.StemmerFactory

C. Feature Extraction (TF-IDF)

Method *Term Frequency-Inverse Document Frequency* (TF-IDF) is used to convert text data into numerical representations [19]. The calculation of weights (TF-IDF) is carried out based on Equation (1):

$$W(dt) = TF(dt) \times \log\left(\frac{N}{DF(t)}\right) + 1$$

Pers.1

Description:
TF(d,t) = the frequency of the occurrence of the word t in the review d.
N = total of all reviews.
DF(t) = the number of reviews that contain the word t.

D. Classification Algorithm Modeling

Comparing the Naive Bayes, Support Vector Machine (SVM) classification algorithm, Random Forest used the 10-Fold Cross-Validation technique by randomly dividing the dataset into 10 equally large folds. In each iteration, 9 folds were used as training data and the remaining 1 fold was used as test data. This process was repeated 10 times so that each review became test data, which aims to avoid bias and prevent overfitting problems.

E. System Performance Evaluation

Performance measurement is carried out by calculating the value of the Confusion Matrix on each fold to determine the Accuracy, Precision, Recall, and F1-Score which groups the prediction results into four categories: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). Calculation of Accuracy, Precision, Recall, and F1-Score, based on these equations

- Accuracy:
  The percentage of the total correct prediction of all the test data on the fold.

$$Akurasi = \frac{\text{Jumlah Prediksi Benar}}{Total\ Data\ Uji\ Ford}$$

Pers.2

- Precision (Presisi):
  Measure the accuracy of the model of all the predicted reviews that go into a class, how many are really accurate.

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

Pers.3

- Recall (Sensitivitas):
  Measuring the model's ability of all reviews that actually belong to a class, how many the model managed to find.

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

Pers.4

- F1 Score:
  A harmonious average between Precision and Recall to measure the balance of model performance (especially if there is an imbalance in the amount of data between classes).
  - $F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$

Pers.5

IV. RESULTS AND DISCUSSION

A. Data Collection

User review data of the Qasir application from the Google Play Store platform uses a data scraping technique based on the Python programming language by utilizing the google-play-scraper library. The use of 2,500 reviews of data that includes userName, star rating, review date (at), and review text (content).

Labeling is carried out based on the user's star rating scale with the following conditions: rating 1–2, categorized as Negative sentiment and rating 3, categorized as Neutral sentiment, and rating 4–5 categorized as Positive sentiment. The results of sentiment labeling can be seen in figure 2.



Figure 2. Positive, Negative and Neutral Sentiment Labeling Results

The results of sentiment labeling show: Positive sentiment as many as 1,425 data and Negative sentiment as many as 825 data and Neutral sentiment as many as 250 data. All data as many as 2,500 lines are stored in ulasan_qasir_2500.csv file.

B. Text Pre-Processing

The scraped review text has a non-standard language structure, contains noise, numbers, punctuation, and spelling variations typical of Play Store users. This stage aims to clean and transform the text data into a structured format that is ready to be processed by the modeling algorithm. 5 stages in pre-processing, namely

1) Cleaning

At this stage, it is done to remove *punctuation*, special symbols, numbers, and emojis so as not to interfere with the feature extraction process.

2) Case Folding

This stage is done to standardize the font format in all comments by changing them to lowercase letters. This is done because not all text documents consistently use capital letters or lowercase letters

3) Tokenization

This stage separates each word by spacing so that each word can be analyzed individually at the stopword and stemming stages.

4) Stopword Removal.

The results of tokenization were analyzed so that the stopwords eliminated common words in Indonesian such as: "yang", "di", "and", "ke".

5) Stemming.

This stage is the last step taken. Stemming changes words that have suffixes, either prefixes, inserts, suffixes to the root word. The results of the pre-processing stages are presented in table III.

TABLE III PRE-PROCESSING RESULTS OF POSITIVE COMMENTS

Stages	Positf Review	Negative Reviews
Comments	Very helpful monthly sales recap!! Steady Qasir, use it simple bgt	Failed to print receipts continue after updating the latest version!! CS is very slow to respond, the app can't be used :(
Cleaning	It's a great way to get a monthly sales rep for Steady Qasir with a simple bgt	Failed to print receipts continue after the latest version of CS slow response the app can't be used
Case Folding	It's a great way to get a monthly recap of the monthly sales rep that is simple bgt	Failed to print receipts continue after updating the latest version of CS The response is very slow the app can't be used
Tokenization	It's very helpful to recap selling the month of Qasir using it is very simple	*failed to print receipts* continue after updating the latest version of customer service slow response very application cannot be used
Stopword Removal	Help recap the sale of the month of Qasir using simple	failed to print receipts update latest version customer service slow response application not used
Stemming	Help recap the sale of the month of Qasir using simple	Failed to print receipts update new version service subscribe slow response application not used

C. Feature Extraction (TF-IDF)

The clean text (*clean content*) resulting from the pre-processing stage is still in the form of a qualitative (string) that cannot be processed directly by machine learning algorithms. Therefore, a feature extraction stage is required to convert the text into a numerical matrix vector form. The feature extraction process is

carried out using *the TfidfVectorizer* function from the *scikit-learn* library in Python.

The results of the Feature Extraction show that words that have specific meanings in Qasir reviews such as "receipt", "print", "fail", or "help" get a fairly high TF-IDF weight in the reviews where the words appear. And words that appear in almost all reviews (such as the word "application" or "qasir") are automatically given a weight of close to zero, so they do not dominate the classification process.

D. Classification Algorithm Modeling

The data was shared and trained using the 10-Fold Cross-Validation technique.

The entire dataset of 2,500 reviews was divided into 10 sections (folds) which were the same size as each fold containing 250 reviews. With training data for each iteration of 90% or 9 fold (2250 reviews) and 10% or 1 fold (250 reviews) as test data. The test results used the 10-Fold Cross-Validation technique, presented in table IV.

TABLE IV. TEST ACCURACY RESULTS PER FOLD (%)

Fold Iteration	SVM (%)	RF (%)	NB (%)
Fold 1	86.40	83.20	77.60
Fold 2	84.80	81.60	75.20
Fold 3	85.60	84.00	78.40
Fold 4	83.60	80.80	74.00
Fold 5	86.80	83.60	78.00
Fold 6	84.40	81.20	76.00
Fold 7	86.00	82.80	79.20
Fold 8	85.20	82.00	75.60
Fold 9	84.00	80.40	73.60
Fold 10	85.20	84.40	78.40
Average	**85.20**	**82.40**	**76.60**

E. System Performance Evaluation

Data distribution through 10-Fold Cross-Validation, the three classification algorithms, namely *Support Vector Machine, Random Forest, and Naïve Bayes, were tested on the TF-IDF weighting matrix to predict sentiment categories (Positive, Negative, Neutral). The evaluation was carried out based on the Confusion Matrix* by calculating the average values and standard deviations on the Accuracy, Precision, Recall, and F1-Score metrics. The code in Python can be seen in Figure 3.

```
import numpy as np
import pandas as pd
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

# Fungsi untuk mencetak dan menyimpan hasil per-fold
def evaluate_models_per_fold(models, X, y, cv):
    results_summary = []

    for name, model in models.items():
        acc_list, prec_list, rec_list, f1_list = [], [], [], []

        for train_idx, test_idx in cv.split(X, y):
            X_train, X_test = X[train_idx], X[test_idx]
            y_train, y_test = y.iloc[train_idx], y.iloc[test_idx]

            # Melatih dan memprediksi
            model.fit(X_train, y_train)
            y_pred = model.predict(X_test)

            # Menghitung metrik per fold
            acc_list.append(accuracy_score(y_test, y_pred))
            prec_list.append(precision_score(y_test, y_pred, average='weighted'))
            rec_list.append(recall_score(y_test, y_pred, average='weighted'))
            f1_list.append(f1_score(y_test, y_pred, average='weighted'))

        # Simpan hasil rata-rata dan deviasi standar
        results_summary.append({
            'Algoritma': name,
            'Akurasi Rata-Rata (%)': np.mean(acc_list) * 100,
            'Precision Rata-Rata (%)': np.mean(prec_list) * 100,
            'Recall Rata-Rata (%)': np.mean(rec_list) * 100,
            'F1-Score Rata-Rata (%)': np.mean(f1_list) * 100,
            'Std Deviasi (±%)': np.std(acc_list) * 100
        })

    return pd.DataFrame(results_summary)

# Eksekusi evaluasi
df_hasil_evaluasi = evaluate_models_per_fold(models, X_tfidf, y, skf)
```

Figure 3. Crawling Confusion Matrix and Std. deviation

The results of the implementation of the code for the system performance for the calculation of the Confusion Matrix and standard deviation, are presented in table V.

TABLE V. CONFUSION MATRIX AND STANDARD DEVIATION

Algorithm	Average Accuracy (%)	Average Precision (%)	Average Recall (%)	Average F1-Score (%)	Standard deviation (±)
SVM	85.20	84.80	85.10	84.95	±1.25
RF	82.40	83.10	81.50	82.29	± 1.80
NB	76.60	75.40	77.20	76.29	±2.40

Based on the results of a 10-fold cross-validation experiment conducted on 2,500 app reviews for Qasir, the Support Vector Machine (SVM) algorithm demonstrated the highest performance, achieving an accuracy of 85.20%, precision of 84.80%, recall of 85.10%, and an F1-score of 84.95%, along with the lowest standard deviation of 1.25%. Both theoretically and empirically, this superior performance of the SVM is driven by several factors related to the model's characteristics and the processing of text data (text mining):

1. Feature extraction using TF-IDF on 2,500 reviews results in a matrix with a high-dimensional feature space (of size M, the vocabulary size). SVM is mathematically designed to perform optimally in high-dimensional vector spaces by identifying an optimal hyperplane that maximizes the separation margin (a "maximum margin classifier"). This makes SVM significantly more robust to the "curse of dimensionality" compared to simpler linear or probabilistic models.

2. SVM focuses solely on the most critical data points, the "support vectors" to determine the decision boundary, thereby avoiding the overfitting often encountered by Random Forest models when dealing with sparse text data.

3. Unlike Naïve Bayes, which assumes that all words are independent of one another (the independence assumption), SVM is able to more effectively capture correlations between words or technical phrases typical of POS reviews.

4. The lowest standard deviation value for SVM (1.25%), compared to Random Forest and Naïve Bayes, indicates that the decision boundary (hyperplane) established by SVM is highly consistent across each fold iteration. The deterministic approach of SVM, based on convex optimization, minimizes error variance, thereby granting the model greater reliability when faced with variations in the distributions of training and test data.

V. CONCLUSION

Based on testing on 2,500 Qasir application review data, it successfully answered the complementary scientific gaps:

A. Comprehensive Multialgorithm Comparison:
Empirically proving the existence of a clear performance hierarchy in the sentiment classification of qasir applications, where Support Vector Machine (SVM) is highest with an average accuracy of 85.20%, and Random Forest has an average accuracy of 82.40%, while Naïve Bayes has 76.60%.
The application of the Stratified 10-Fold Cross-Validation scheme and the integration of TF-IDF weights, were able to eliminate the bias of sharing a single random data that was often a limitation in previous Qasir review research.

B. Full Metric-Based Evaluation:
The test not only measured Accuracy, but also proved the balance of Precision (84.80%), Recall (85.10%), and F1-Score (84.95%) in SVM. In addition, standard deviation measurements confirmed that SVM had the highest stability, compared to Random Forest and Naïve Bayes.

Overall, the integration of TF-IDF and Stratified 10-Fold Cross-Validation with the Support Vector Machine (SVM) algorithm proved to be the most accurate, stable, and reliable solution for analyzing user sentiment on the Qasir digital cashier application domain.

ACKNOWLEDGMENTS

Thank you to the Chairman of LPPM IBN for providing academic support and facilities, as well as the opportunity for the Author to be able to complete this research

REFERENCES

[1] A. Emier *dkk.*, "The Application of Web-Based Cashier Applications to Optimize Transaction Services for MSMEs," *RIGGS*, vol. 4, no. 4, pp. 15076–15082, Jan 2026, doi: 10.31004/riggs.v4i4.5546.

[2] A. I. Nababan, A. H. Nababan, R. M. Afifa, and R. Hikmah, "Systematic Literature Review: Adoption of Point of Sales (POS) Applications in Improving MSME Performance in Indonesia," *EKOMA : Journal of Economics, Management, Accounting*, vol. 5, no. 5, 2026.

[3] F. A. Mujtahid, B. Zaman, and G. G. Aji, "Analysis of Neobank Sentiment Classification: Comparison of N-Gram Configurations on TF-IDF Using Naive Bayes and SVM," *tin*, vol. 6, no. 12, pp. 2375–2383, May 2026, doi: 10.47065/tin.v6i12.9702.

[4] R. Mursyid and A. D. Indriyanti, "Comparison of the Accuracy of Sentiment Analysis Methods for User Opinion Evaluation on Social Media Platforms (Case Study: Twitter)," *JINACS: Journal of Informatics and Computer Science*, vol. 06, no. 02, 2024.

[5] C. Cantika, M. D. Sitompul, and A. Wijaya, "Analysis of App Reviews in the Google Play Store Using the Naive Bayes Model," *Journal of Educational Multidisciplinary Research*, vol. 3, no. 1, pp. 1–17, 2026.

[6] M. A. Ramadhani *et al.*, "Implementation of Support Vector Machine (SVM) Algorithm for Web Application-Based Human Health Diagnosis," *Jurnal Ners Research & Learning in Nursing Science*, vol. 9, no. 1, pp. 896–902, 2025.

[7] I. S. Aisah, B. Irawan, and T. Suprapti, "Support Vector Machine (SVM) Algorithm for Sentiment Analysis of Digital Quran Application Reviews," *JATI (Student Journal of Informatics Engineering)*, vol. 7, no. 6, 2023.

[8] R. Fachuzi and R. Vindua, "Application of Support Vector Machine (SVM) for Hoax Detection in Online News," *JATI (Student Journal of Informatics Engineering)*, vol. 10, no. 4, 2026.

[9] E. M. S. Sakti, K. Diantoro, Marnis, and Tatiyani, "Development of an Artificial Intelligence-Based MSME Economic Decision Model Using a Cognitive Computing Approach," *Journal of Infokom Essence*, vol. 10, no. 1, 2026.

[10] L. Sinaga, M. S. Batubara, and N. S. Siagian, "Performance Analysis of Random Forest Algorithm in Student Graduation Classification Using Student Performance Dataset," *Ikraith Informatics*, vol. 10, no. 2, 2026, doi: https://doi.org/10.37817/ikraith-informatika.v10i2.

[11] P. S. Saputra and I. P. G. A. Sudiatmika, "Analysis of Smartphone Price Prediction in 2023 Using Random Forest Regression Model Based on Technical Specification Features," *Journal of Computer Science and Technology (KOMTEKS)*, vol. 3, no. 2, pp. 13–17, 2024.

[12] A. Haris, A. Hananto, and F. Nurapriani, "Application of Naive Bayes Algorithm for Sentiment Classification of User Reviews of Pidi MSME Application with TF-IDF Feature Weighting," *JATI (Student Journal of Informatics Engineering)*, vol. 9, no. 5, 2025.

[13] D. W. Putri and M. A. Soeleman, "The Application of the Naïve Bayes Algorithm to the Sentiment of Skincare Product Reviews on Shopee E-Commerce," *Building of Informatics, Technology and Science (BITS)*, vol. 7, no. 4, 2026.

[14] Suprapto, "Improvement Naive Bayes Using Forward Selection, Information Gain and Gain Ratio for Feature Independence Treatment," *Journal of Social and Technology (SOSTECH)*, vol. 5, no. 4, 2025.

[15] A. Sari, B. Irawan, A. Faqih, A. R. Dikananda, and F. Fathurrohman, "Sentiment Analysis of Kredivo Application User Reviews Using the Support Vector Machine (SVM) Algorithm with the TF–IDF Method," *LINEAR*, vol. 2, no. 4, pp. 593–601, Feb 2026, doi: 10.33096/linier.v2i4.3344.

[16] M. R. Mahrifan and F. Yunita, "Sentiment Analysis on MyXL Application Using Naïve Bayes Algorithm," *Ice cream*, vol. 4, no. 3, 2026.

[17] L. Hartimar, Y. Manza, dan K. P. Siregar, "Text Classification Using TF-IDF and Naïve Bayes: Case Study of MyXL App User Review Data," *Journal of Technology and Computer (JOTECHCOM)*, vol. 2, no. 2, pp. 100–108, 2025.

[18] M. H. H. Habibi, E. D. Wahyuni, and R. Permatasari, "Comparison of SVM and RF Algorithm Performance in Sentiment Classification with Sarcasm Detection on Youtube Comments," *JATI (Student Journal of Informatics Engineering)*, vol. 9, no. 2, 2025.

[19] M. D. Bimantara dan I. Zufria, "Text Mining Sentiment Analysis on Mobile Banking Application Reviews using TF-IDF Method with Natural Language Processing Approach," *Jinav J. Inf. Vis.*, vol. 5, no. 1, pp. 115–123, Jul 2024, doi: 10.35877/454RI.jinav2772.