Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

15th February 2025. Vol. 4. No. 2; e-ISSN: 2808-4519

Comparison of Sentiment Analysis Models Enhanced by Naïve Bayes and Support Vector Machine Algorithms on Mobile Banking BRImo Reviews

Muhamad Firly Ramadan[1]*, Martanto[2], Arif Rinaldi Dikananda[3], Ahmad Rifa'i[4]

[1,2,3,4] *STMIK IKMI Cirebon*

muhamadfirlyramadan@gmail.com[1*], martantomusijo@gmail.com[2], rinaldi21crb@gmail.com[3], a.rifaaii1408@gmail.com[4]

Abstract

This study compares the effectiveness of the Support Vector Machine (SVM) and Naïve Bayes algorithms in classifying user sentiment regarding the BRImo application. User reviews were obtained from the Google Play Store platform and underwent a text preprocessing stage to clean and prepare the data. Subsequently, the SVM and Naïve Bayes algorithms were applied for sentiment analysis, using evaluation metrics such as accuracy, precision, recall, and F1-score. The results show that SVM achieved a training accuracy of 95.67% and a testing accuracy of 83.11%, with its best performance on positive sentiment (precision 92.26%, recall 91.79%, F1-score 92.02%) and moderate performance on negative sentiment (precision 62.81%, recall 62.81%, F1-score 62.81%). Meanwhile, Naïve Bayes recorded a training accuracy of 95.23% and a testing accuracy of 82.77%, with its highest performance on positive sentiment (precision 90.12%, recall 93.38%, F1-score 91.72%) but lower performance on negative sentiment (precision 65.07%, recall 60.06%, F1-score 62.46%). In terms of sentiment distribution, SVM was more effective in handling sentiment variations, particularly in detecting negative and neutral sentiments. These findings indicate that SVM outperforms Naïve Bayes in sentiment analysis of user reviews for the BRImo application.

Keywords: *Naïve Bayes, Support Vector Machine, Machine Learning, KDD, Brimo, Sentiment Analysis, Mobile Banking.*

1. Introduction

The BRImo mobile banking application has become an integral part of daily life, allowing users to manage financial transactions effortlessly. With an increasing user base, reviews have become a critical source of information for developers to identify user complaints or feedback [1]. However, processing and analyzing these reviews effectively poses a challenge. Sentiment analysis is a primary method for understanding user perceptions, and this research aims to compare the Naïve Bayes and Support Vector Machine (SVM) algorithms to enhance opinion analysis models on BRImo application reviews.

Support Vector Machine (SVM) and Naïve Bayes are two widely-used approaches in opinion analysis, frequently utilized to interpret sentiment patterns in textual data. These methods have distinct characteristics but are both effective in processing text-based reviews or opinions. SVM excels in separating high-dimensional data with high accuracy, while Naïve Bayes is renowned for its simplicity and efficiency. Although these methods have been extensively applied in sentiment analysis across various domains, studies comparing their performance in the context of mobile banking reviews remain limited. This research aims to evaluate the effectiveness of both algorithms in optimizing sentiment analysis models for BRImo application reviews.

Previous studies demonstrate the successful application of Naïve Bayes and SVM in sentiment analysis. Research by reference [2] employed the Naïve Bayes algorithm to analyze public sentiment about COVID-19, achieving 78% accuracy. In the agricultural sector, reference [3] used Naïve Bayes for diagnosing corn plant diseases with 90% accuracy. Meanwhile, reference [4] applied SVM to analyze mental pressure sentiments on Twitter, attaining a high accuracy of 99.34%. These findings encourage further research to apply both methods to mobile banking review analysis.

This study will collect user reviews of BRImo and utilize Python along with development environments like Google Colab. Naïve Bayes and SVM algorithms will be implemented using the scikit-learn library, and their performance will be compared using evaluation metrics such as accuracy, precision, and recall. The study aims to determine which algorithm is superior in optimizing sentiment analysis models for BRImo application reviews.

By comparing the performance of Naïve Bayes and SVM algorithms, this study seeks to provide comprehensive guidance for mobile banking application developers, such as those of BRImo, in selecting the most suitable algorithm to enhance service quality through user sentiment analysis. The findings of this research are also expected to enrich insights into the application of sentiment analysis in the

financial application context. Overall, this study significantly contributes to the field of informatics, particularly in the development of more reliable machine learning techniques for sentiment analysis of mobile banking reviews. The practical guidance produced can assist application developers in improving services based on user feedback.

1.1. Naïve Bayes

Naïve Bayes is a highly efficient linear classifier. The Naïve Bayes probabilistic model and classification approach are based on Bayes theorem, with the term "naïve" stemming from the assumption that the features in the dataset are mutually independent [5]. Naïve Bayes is a method used to predict probabilities and possesses several characteristics that intuitively align with the reasoning of an expert, supported by a strong mathematical foundation. [3].

1.2. Support Vector Machine

Support Vector Machine (SVM) is a machine learning algorithm used to address classification and regression problems by mapping data into a higher-dimensional space to find the optimal hyperplane that separates data classes with the maximum margin. This algorithm works by identifying key data points, called support vectors, which help determine the decision boundary (hyperplane). For data that cannot be linearly separated, SVM employs the kernel trick approach to transform the data into a dimension where better separation is possible.

1.3. Knowledge Discovery Database

Knowledge Discovery in Database (KDD) is a method used to obtain information or knowledge contained in an existing database, which consists of interconnected tables. The results of this knowledge discovery process can serve as a knowledge base to support decision-making. KDD and data mining are often used to extract hidden information from large-scale databases. The KDD process consists of five stages: data selection, preprocessing, transformation, data mining, and evaluation [6].

2. Research Method

2.1. Research Method

The methodology applied in this study is Knowledge Discovery in Databases (KDD). The process or research workflow followed in this study is illustrated in Figure 1 below:

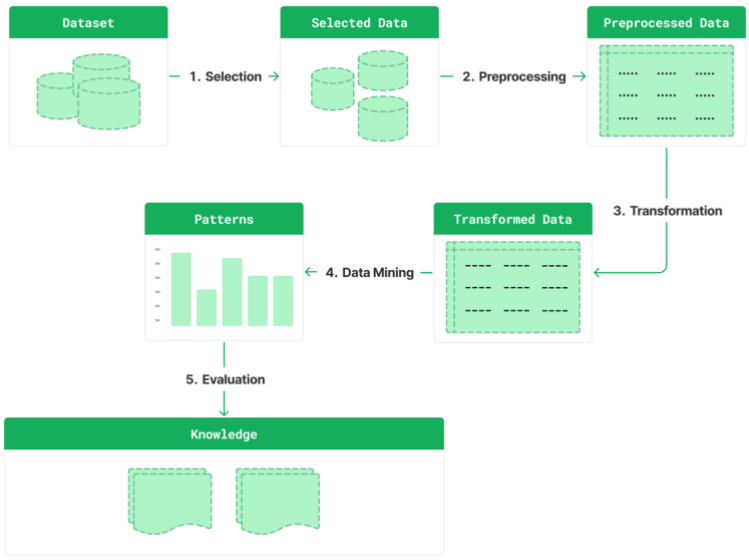

Fig. 1: Research Diagram.

3. Result And Discussion

3.1. Selection

The dataset used consists of user reviews for the BRImo application, collected from the Google Play Store platform and stored in Comma Separated Values (CSV) file format. The data collection process was conducted using web scraping techniques.

Table 1: BRImo Reviews Dataset.

Review Id	userName	Content	At
782b1e99-7a55-4657-9966-9d0c07570a61	Ana Diana	Good	2024-08-31 23:59:49

2966eb1e-23dd-4508-83bf-cf590df5a304	Anisa Awali	Mantullllll	2024-08-31 23:56:40
b8e8c4bc-966e-40cd-a5c7-8a117628b56f	Tanti Siahaya	BRImo makes it very easy for me to make transactions, but the admin fee for mobile credit and token transactions is too expensive. Maybe the admin fee could be reduced even more. BRImo 😊😊🙏	2024-08-31 23:55:49
7dbc2373-e2da-4881-975c-e230af013bea	Rahmat Ahmat	Very Good	2024-08-31 23:50:23
8335eabe-5b5f-4b29-be36-21fbdd03181c	angga risbiantoro	Very Helpfull	2024-08-31 23:48:59

3.2. Preprocessing

The first stage in preprocessing is text cleaning. One of the processes implemented in text cleaning is case folding, where all uppercase letters are converted to lowercase. This aims to standardize words that have the same meaning but are written with different capitalizations. Below is an example of the result from the text cleaning process presented in the table below.

Table 2: Result of Text Cleaning Process.

Input Text	Output Text
How do I register? Do I need to go to the nearest BRI office? I have already uploaded my ID card photo and entered my details online, but when I click continue, it says that my phone number is not registered.	how to register do i need to go to the nearest bri office because i have already uploaded the id card photo entered the data but when i click continue there is a message saying your phone number is not registered
BRImo makes it very easy for me to make transactions, but for mobile credit and token transactions, the admin fee is too expensive. Maybe the admin fee could be reduced further. BRImo 😊😊🙏	brimo makes it very easy for me to make transactions but for mobile credit and token transactions the admin fee is too expensive maybe the admin fee could be reduced further brimo

After the text is cleaned, the next step is stopwords removal. Stopwords removal is the process of eliminating words that are not necessary because they do not carry significant meaning in the preprocessing phase. The goal of the stopwords process is feature selection, which aims to perform weighting to create a dataset [4]. Below is an example of the result from the stopwords removal phase presented in Table 3 below.

Table 3: Result of Stopword Removal.

Input Text	Output Text
let the stars speak	let stars speak
app fail login continuously	app fail
very simplify matters	simplify matters

After stopwords removal, the correct spelling process will be applied to the content column. This process aims to correct spelling errors that may interfere with understanding during data analysis, as well as the use of slang and non-standard terms. Below is an example of the result from the correct spelling phase presented in Table 4 below.

Table 4: Result of Correct Spelling.

Input Text	Output Text
mantullllll	very good
memebantu	helpfull
app fail	application failed

After correcting the spelling and applying additional stopwords removal, the next step is stemming. Stemming is the process of selecting words with prefixes, suffixes, pronouns, and verbs to convert them into their base form by removing prefixes or suffixes. For example, "Bersama melawan virus corona" is transformed into "sama lawan virus corona" [7]. Below is an example of the result from the stemming phase presented in the table below.

Table 5: Result of Stemming Process.

Input Text	Output Text
helpfull	help
facilitate affairs	easy affair
facilitates transactions	easy transaction

The remove missing value process is a crucial stage in dataset preparation. This stage aims to ensure that the data used in the research does not contain missing values (NaN), which could significantly affect the analysis results. The number of NaN values in the dataset can be seen in the image below.

```
print(f"Count of NaN content in dataset: {len(nan_content)}")

Count of NaN content in dataset: 1180
```

Fig 2: Count of NaN Value in Dataset.

Figure 3 shows the code used to provide quantitative information about the number of missing values. It was found that there are 1180 data points containing NaN values. After identifying and determining the number of NaN values, the next step is to remove the content containing NaN values from the dataset. After handling the missing values in the dataset, the next step is to perform the tokenization process. Tokenization is the process of breaking down a sentence in a paragraph into individual words that are not related to each other, resulting in standalone words [4].The result of the tokenization process is shown in the image below.

Table 6: Result of Tokenization Process.

Input Text	Output Text
helpfull	[help]
facilitate affairs	[easy, affair]
facilitates transactions	[easy, transaction]

Table 3 shows the result of tokenization in the cleaned_content column, indicating that each row of text has been broken down into a collection of words or tokens, stored in the tokens column. After applying the tokenization process to the DataFrame, the next step is the labeling process. Labeling is a crucial step in sentiment analysis to assign labels (positive, negative, or neutral) to each piece of text. With IndoBERT, labeling is performed automatically by analyzing the sentiment score (threshold) of the text. IndoBERT is a transformer-based model that follows the BERT style and is trained as a masked language model using the Huggingface framework [8], [9]. The result of the labeling phase can be seen in the table below.

Table 7: Result of Labeling Process.

Input Text	Label
good	Positive
very good	Positive
help	Neutral
easy i transact but transaction credit token expensive admin fee reduce admin fee credit and token too expensive admin fee maybe can reduce again brimo	Negative

3.3. Transformation

At this stage, the data in the cleaned_content column, which has undergone the preprocessing process, is transformed into vectors using the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization technique. TF-IDF is a process that transforms text data into numerical data for weighting each word in each document. TF represents the frequency of a word appearing in a given document, indicating how important and significant that word is within the document. DF is the frequency of documents containing that word, representing how common that word is. IDF is the inverse of DF [4].The resulting feature vectors can be seen in the image below.

	aaah	aaah gelapp	aamiin	aamiin istajib	aamiin terima	aamiin yra	aauh	aauh aaah	abdet	abdet nama	...	za	zaman	zaman digital	zoonk	zoonk sya	zte	zte blade	zzz	zzz xyxy	zzz
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4934	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4935	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4936	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4937	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4938	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

12347 rows × 16208 columns

Fig. 3: Result of Vectorization Process.

After transforming the text data into vectors using the TF-IDF technique, the dataset is divided into a training set and a test set using the train_test_split function from scikit-learn, with a proportion of 60% training data and 40% test data. To address the imbalance in the training data, the Synthetic Minority Oversampling Technique (SMOTE) is applied. The result of the SMOTE process can be seen in the image below.

label	count
1	4440
2	4440
0	4440

dtype: int64

Fig. 4: Train set after SMOTE.

After SMOTE is applied, the previously imbalanced training data now has an even distribution across each class. This process is crucial to ensure that the model can learn patterns from each class proportionally.

3.4. Data Mining.

At this stage, cross-validation is applied to optimize the hyperparameters of the Support Vector Machine (SVM) and Naïve Bayes models using a 10-fold cross-validation scheme. The cross_val_score function from the scikit-learn library is used for consistent model evaluation and to avoid overfitting. The SVM modeling uses the SVC algorithm, with hyperparameter optimization through GridSearchCV, which includes the parameters C, kernel, gamma, and class_weight with the following grid.

Table 8: Params Grid SVM Model.

C	*kernel*	*gamma*	*class_weight*
[0.1, 1, 10]	[linear,rbf]	[scale,auto]	[balanced,None]

The results of the parameter search for the Support Vector Machine model can be seen in the image below.

```
Best parameters found:  {'C': 10, 'class_weight': 'balanced', 'gamma': 'scale', 'kernel': 'rbf'}
Best cross-validation accuracy: 95.67%
```

Fig. 5: GridSearch result SVM Model.

The results of the parameter search using GridSearchCV show the best parameters: C=10, class_weight='balanced', gamma='scale', and kernel='rbf', with a cross-validation accuracy on the training data of 95.67%. Meanwhile, the Naïve Bayes modeling uses the Multinomial Naïve Bayes algorithm with parameter optimization for alpha (for Laplace smoothing) and fit_prior (for class prior distribution), with the following grid.

Table 9: Params Grid Naive Bayes Model.

alpha	*fit_prior*
[0.01, 0.1, 1, 10]	[True, False]

The results of the parameter search for the Naïve Bayes model can be seen in the image below.

```
Best parameters found:  {'alpha': 0.01, 'fit_prior': True}
Best cross-validation accuracy: 95.23%
```

Fig. 6: GridSearch result Naive Bayes Model.

The optimization results show the best parameters: alpha=0.01 and fit_prior=True, with a cross-validation accuracy on the training data of 95.23%. This approach ensures optimal and consistent model performance for sentiment analysis on BRImo mobile banking app reviews.

3.5. Evaluation & Interpretation

After the implementation of the Support Vector Machine (SVM) and Naïve Bayes algorithms, the next step is the interpretation and evaluation of the results. Evaluation is performed using metrics such as accuracy, precision, recall, and F1-score to measure the model's performance in classifying sentiment.

The Confusion Matrix generated by the Support Vector Machine model for the classification applied to the test (train set) can be seen in the image below.

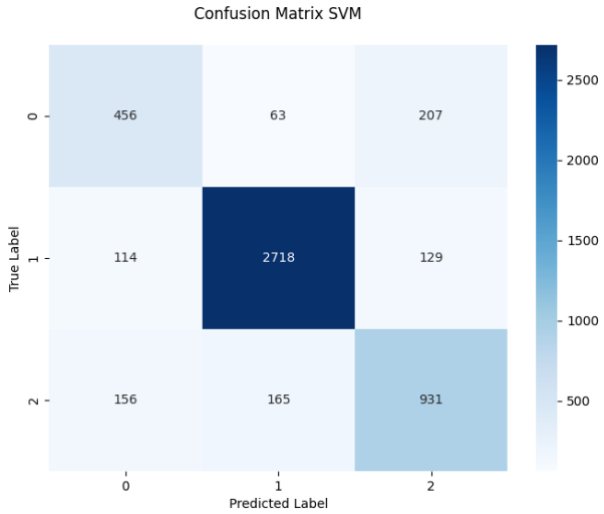

Fig. 7: Confusion Matrix SVM Model.

Based on the confusion matrix shown in the image above, the following can be observed:

1. *Accuracy*

   True Positive (TP) total = 456 + 2718 + 931 = 4,105

   Total Data = 456 + 2718 + 931 + 270 + 228 + 336 = 4,939

   Thus:

$$Accuracy = \frac{True\ Positive\ Total}{Total\ Data} = \frac{4{,}105}{4{,}939} = 0.8311$$

2. *Class* 1 (*Positive*) *precission*, *recall* and *F1-Score:*

   Given:

   TP = 2718 FP = 228

   FN = 243

$$Precision = \frac{TP}{TP + FP} = \frac{2718}{2718 + 228} = 0.9226$$

$$Recall = \frac{TP}{TP + FN} = \frac{2718}{2718 + 243} = 0.9179$$

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

$$F1 - score\ = 2\ \times \frac{0.9226\ \times\ 0.9179}{0.9226 + 0.9179} = 0.9202$$

3. *Class* 0 *(Negative) precission*, *recall* and *F1-Score*:

   Given:

   TP = 456 FP = 270

   FN = 270

$$Precision = \frac{TP}{TP + FP} = \frac{456}{456 + 270} = 0.6281$$

$$Recall = \frac{TP}{TP + FN} = \frac{456}{456 + 270} = 0.6281$$

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

$$F1 - score = 2\ \times \frac{0.6281\ \times\ 0.6281}{0.6281 + 0.6281} = 0.6281$$

4. *Class* 2 *(Neutral) precission, recall* and *F1-Score*:

   Given:

   TP = 931 FP = 336

   FN = 321

$$Precision = \frac{TP}{TP + FP} = \frac{931}{931 + 336} = 0.7348$$

$$Recall = \frac{TP}{TP + FN} = \frac{931}{931 + 336} = 0.7436$$

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

$$F1 - score = 2 \times \frac{0.7348 \times 0.7436}{0.7348 + 0.7436} = 0.7392$$

Based on the analysis of the confusion matrix for the SVM model, the evaluation results show a test accuracy of 83.11%. The confusion matrix analysis indicates that the model made 4,105 correct predictions out of a total of 4,939 data points, with the best precision, recall, and F1-score in the positive (1) class, which are 92.26%, 91.79%, and 92.02%, respectively. However, for the negative (0) and neutral (2) classes, the model still exhibits weaknesses, with a precision and recall of 62.81% for the negative class and an F1-score of 73.92% for the neutral class.

Meanwhile, the Confusion Matrix generated by the Support Vector Machine model for the classification applied to the test (train set) can be seen in the image below.

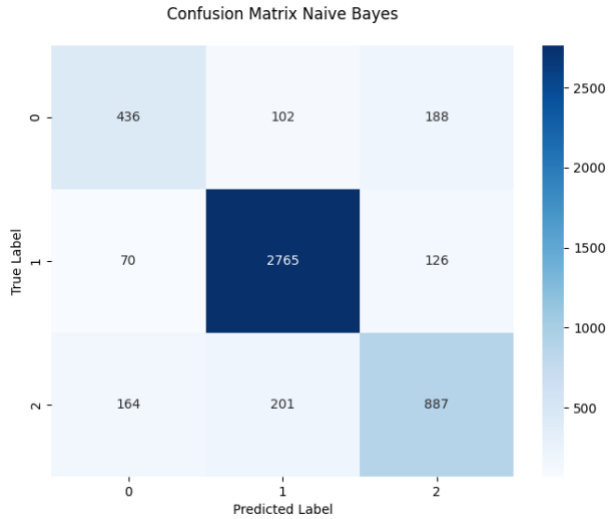

Fig. 8: Confusion Matrix Naive Bayes Model.

Based on the confusion matrix shown in the image above, the following can be observed:

1. *Accuracy*

   True Positive (TP) total = 436 + 2765 + 887 = 4,088

   Total Data = 436 + 2765 + 887 + 303 + 314 + 234 = 4,939

   Thus:

$$Accuracy = \frac{True\ Positive\ (TP)\ Total}{Total\ Data} = \frac{4{,}088}{4{,}939} = 0.8277$$

2. *Class* 1 (*Positive*) *precission, recall* and *F1-Score:*

   Given:

   TP = 2,765 FP = 303

   FN = 196

$$Precision = \frac{TP}{TP + FP} = \frac{2765}{2765 + 303} = 0.9012$$

$$Recall = \frac{TP}{TP + FN} = \frac{2765}{2765 + 196} = 0.9338$$

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

$$F1 - score = 2 \times \frac{0.9012 \times 0.9338}{0.9012 + 0.9338} = 0.9172$$

3. *Class* 0 *(Negative) precission, recall* and *F1-Score*:

   Given:

   TP = 436 FP = 234

   FN = 290

$$Precision = \frac{TP}{TP + FP} = \frac{436}{436 + 234} = 0.6507$$

$$Recall = \frac{TP}{TP + FN} = \frac{436}{436 + 290} = 0.6006$$

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

$$F1 - score = 2 \times \frac{0.6507 \times 0.6006}{0.6507 + 0.6006} = 0.6246$$

4. *Class* 2 *(Neutral) precission, recall* and *F1-Score*:

   Given:

   TP = 887 FP = 314

   FN = 365

$$Precision = \frac{TP}{TP + FP} = \frac{887}{887 + 314} = 0.7386$$

$$Recall = \frac{TP}{TP + FN} = \frac{887}{887 + 365} = 0.7085$$

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

$$F1 - score = 2 \times \frac{0.7386 \times 0.7085}{0.7386 + 0.7085} = 0.7232$$

Based on the analysis of the confusion matrix for the Naïve Bayes model, it shows a train accuracy of 95.23% and a test accuracy of 82.77%. The confusion matrix analysis indicates a total of 4,088 correct predictions out of 4,939 data points. The model performs best on the positive (1) class with a precision of 90.12%, recall of 93.38%, and an F1-score of 91.72%, while the performance on the negative (0) and neutral (2) classes is less optimal, with a precision of 65.07% for the negative class and an F1-score of 72.32% for the neutral class.

Based on the analysis results, both algorithms perform well in recognizing positive reviews, but the SVM model overall shows higher accuracy and evaluation metrics compared to the Naïve Bayes model. Nevertheless, both models still have room for improvement, particularly in classifying negative and neutral sentiment reviews.

Referring to the analysis results, the sentiment distribution generated by the Support Vector Machine (SVM) algorithm on the BRImo mobile banking app review dataset shows that user reviews are divided into three sentiment categories: Positive, Neutral, and Negative.

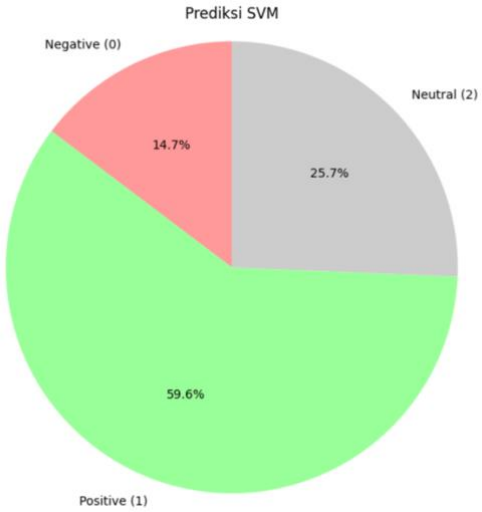

Fig. 9: Sentiment Distribution SVM Model.

Based on the image 11 above, it can be seen that the majority of reviews fall into the Positive category, accounting for 59.6%. Next, Neutral reviews make up 25.7% of the total data, while Negative reviews account for only 14.7%. The sentiment distribution generated by the Naïve Bayes algorithm on the BRImo mobile banking app review dataset shows that user reviews are divided into three sentiment categories: Positive, Neutral, and Negative.

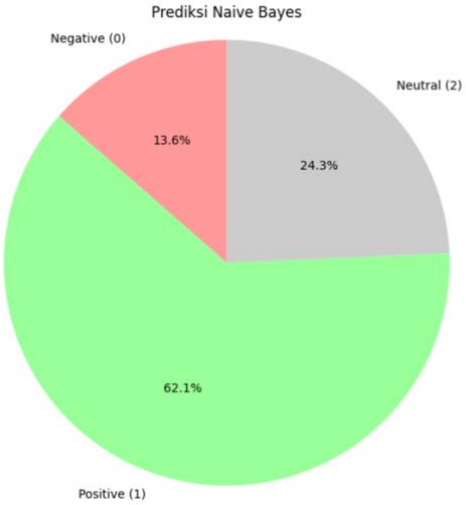

Fig. 10: Sentiment Distribution Naive Bayes Model.

Based on the image 4.59 above, it can be seen that the Positive category has the largest proportion, which is 62.1%. The Neutral category accounts for 24.3% of the total data, while the Negative category has a proportion of 13.6%.

This research supports the findings of several previous studies on sentiment analysis and the effectiveness of the Naïve Bayes and Support Vector Machine (SVM) algorithms. A study by reference [10] showed that SVM had the best overall performance, although Naïve Bayes recorded the highest precision at 83.81%. This finding aligns with the current research, where SVM excels in accuracy (83.11%), precision, and recall, especially in detecting negative and neutral sentiments.

Reference [1] found that Naïve Bayes achieved an accuracy of 84.52% in analyzing BRImo reviews, but this research shows that SVM is superior in detecting neutral and negative sentiments. Another study by reference [11] also indicates the superiority of SVM with an accuracy of 79.5% compared to Naïve Bayes (72.5%) in the context of sentiment analysis related to the economic recession.

The study by reference [12] noted that Naïve Bayes had an accuracy of 85.06% on logistics company reviews, while reference [13] showed an accuracy of 74% on imbalanced datasets. These findings highlight that dataset quality significantly impacts model performance. In this study, the large dataset (4,939 test data) allowed SVM to show better performance than Naïve Bayes.

The majority of BRImo reviews are positive (59.6%), followed by neutral (25.7%) and negative (14.7%). This finding is consistent with reference [1], who noted that the majority of reviews were positive (53.09%). This suggests a positive user perception of the BRImo application, with minimal negative reviews.

4. Conclusion

Based on the research results, the Support Vector Machine (SVM) algorithm proves to be superior to Naïve Bayes in sentiment analysis of BRImo application reviews. SVM achieved an average train accuracy of 95.67% and a test accuracy of 83.11%, slightly higher than Naïve Bayes, which obtained a train accuracy of 95.23% and a test accuracy of 82.77%. SVM also demonstrated more stable performance in detecting sentiments across various categories.

For negative sentiment, SVM recorded precision and recall of 62.81%, compared to Naïve Bayes, which had precision of 65.07% and recall of 60.06%. For positive sentiment, SVM had precision of 92.26% and recall of 91.79%, slightly better than Naïve Bayes, which had precision of 90.12% and recall of 93.38%. In the classification of neutral sentiment, SVM achieved precision of 73.48% and recall of 74.36%, surpassing Naïve Bayes with precision of 73.86% and recall of 70.85%.

Additionally, sentiment distribution shows that SVM is more reliable in distinguishing between neutral and negative reviews, classifying 59.6% of the data as positive, 14.7% as negative, and 25.7% as neutral. Naïve Bayes classified 62.1% of the data as positive, 13.6% as negative, and 24.3% as neutral. Overall, SVM provides better and more consistent performance in the sentiment analysis of BRImo application reviews.

References

[1] M. Khoirul, U. Hayati, and O. Nurdiawan, "Analisis Sentimen Aplikasi Brimo Pada Ulasan Pengguna Di Google Play Menggunakan Algoritma Naive Bayes," 2023.

[2] E. T. Handayani and A. Sulistiyawati, "Analisis Sentimen Respon Masyarakat Terhadap Kabar Harian Covid-19 Pada Twitter Kementerian Kesehatan Dengan Metode Klasifikasi Naive Bayes," *Jurnal Teknologi dan Sistem Informasi (JTSI)*, vol. 2, no. 3, pp. 32–37, 2021, [Online]. Available: http://jim.teknokrat.ac.id/index.php/JTSI

[3] B. B. Suherman, "Sistem Pakar Diagnosa Penyakit Dan Hama Pada Tanaman Jagung Menggunakan Metode Naive Bayes," *Jurnal Informatika dan Rekayasa Perangkat Lunak (JATIKA)*, vol. 2, no. 3, pp. 390–398, 2021, [Online]. Available: http://jim.teknokrat.ac.id/index.php/informatika

[4] D. Atika, A. Ari Aldino, S. Informasi, J. Pagar Alam No, L. Ratu, and K. Kedaton, "Term Frequency-Inverse Document Frequency Support Vector Machine Untuk Analisis Sentimen Opini Masyarakat Terhadap Tekanan Mental Pada Media Sosial Twitter," 2022. [Online]. Available: http://jim.teknokrat.ac.id/index.php/JTSI

[5] A. Muzaki and A. Witanti, "Sentiment Analysis Of The Community In The Twitter To The 2020 Election In Pandemic Covid-19 By Method Naive Bayes Classifier," *Jurnal Teknik Informatika (Jutif)*, vol. 2, no. 2, pp. 101–107, Mar. 2021, doi: 10.20884/1.jutif.2021.2.2.51.

[6] S. N. Rismanah, R. Astuti, and F. M. Basysyar, "Penerapan Algoritma Support Vector Machine Dalam Menganalisis Sentimen Ulasan Pelanggan Shopeefood Berdasarkan Twitter," 2024.

[7] N. Satya Marga, A. Rahman Isnain, and D. Alita, "Sentimen Analisis Tentang Kebijakan Pemerintah Terhadap Kasus Corona Menggunakan Metode Naive Bayes," *Abstrak*, vol. 453, no. 4, pp. 453–463, 2021, [Online]. Available: http://jim.teknokrat.ac.id/index.php/informatika

[8] J. Devlin, M.-W. Chang, and K. Lee, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." [Online]. Available: https://github.com/tensorflow/tensor2tensor

[9] F. Koto, A. Rahimi, J. H. Lau, and T. Baldwin, "IndoLEM and IndoBERT: A Benchmark Dataset and Pre-trained Language Model for Indonesian NLP." [Online]. Available: https://huggingface.co/

[10] R. T. Aldisa and P. Maulana, "Analisis Sentimen Opini Masyarakat Terhadap Vaksinasi Booster COVID-19 Dengan Perbandingan Metode Naive Bayes, Decision Tree dan SVM," *Building of Informatics, Technology and Science (BITS)*, vol. 4, no. 1, pp. 106–109, Jun. 2022, doi: 10.47065/bits.v4i1.1581.

[11] S. A. Sutresno, "Analisis Sentimen Masyarakat Indonesia Terhadap Dampak Penurunan Global Sebagai Akibat Resesi di Twitter," *Building of Informatics, Technology and Science (BITS)*, vol. 4, no. 4, Mar. 2023, doi: 10.47065/bits.v4i4.3149.

[12] I. R. Afandi, F. Noor, H. #2, A. A. Rizki, N. Pratiwi, and Z. Halim, "Analisis Sentimen Opini Masyarakat Terkait Pelayanan Jasa Ekspedisi Anteraja Dengan Metode Naive Bayes." [Online]. Available: https://t.co/2HAdwg1drL

[13] Fitri Wulandari, Elin Haerani, Muhammad Fikry, and Elvia Budianita, "Analisis sentimen larangan penggunaan obat sirup menggunakan algoritma naive bayes classifier," *Jurnal CoSciTech (Computer Science and Information Technology)*, vol. 4, no. 1, pp. 88–96, May 2023, doi: 10.37859/coscitech.v4i1.4781.