COMPARISON OF NBC, SVM, KNN CLASSIFICATION RESULTS IN SENTIMENT ANALYSIS OF MOBILE JKN

Nadya Bethry Balqies Tjikdaphia¹*, Sulastri¹
¹Fakultas Teknologi Informasi dan Industri, Universitas Stikubank
email: *nadyabethrybalqiestjikdaphia@mhs.unisbank.ac.id

Abstract: The JKN Mobile application is a mobile application created to facilitate healthcare administration in Indonesia since 2017. The application has been downloaded by over 10 million users and has received 484,000 diverse reviews, including positive, negative, and neutral feedback. The average rating given by users is 4.5 out of 5 stars. This research aims to perform sentiment analysis on user reviews found in the Google Play Store review column. The methods used for sentiment analysis are Naive Bayes, K-Nearest Neighbor (K-NN), and Support Vector Machine (SVM). The test results show that with a 10% test data and 90% training data proportion, the SVM method achieves the highest accuracy of 95%. Naive Bayes follows with an accuracy of 87%, and K-NN with an accuracy of 75%.

Keywords: JKN mobile application, sentiment analysis, naïve bayes, k-nearest neighbor (K-NN), support vector machine (SVM).


Kata kunci: JKN mobile, analisis sentimen, naïve bayes, k-nearest neighbor (K-NN), support vector machine (SVM).
INTRODUCTION

In this era of globalization, BPJS Kesehatan has developed a mobile application called JKN Mobile to facilitate users in health registration and management. This application has been introduced since 2017 [1] and has more than 10 million users with an average rating of 4.5 stars out of 5 stars and a total of 513 thousand reviews [2]. The aim of this research is to analyze the sentiment of JKN Mobile application reviews on the Google Play Store using the Naive Bayes, K-Nearest Neighbor (K-NN), and Support Vector Machine (SVM) methods.

Naive Bayes classification is one of the classification algorithms based on Bayes’ theorem with the assumption of independence between the existing features. This algorithm is very popular because it is simple, efficient in resource usage, and able to provide good results in many applications [3]. Support Vector Machine (SVM) is a machine learning algorithm used for classification and regression tasks. SVM works by building a model that can separate two classes by finding the best hyperplane that has the maximum margin between the classes [4]. K-Nearest Neighbor (K-NN) is a machine learning algorithm used for classification and regression tasks. KNN works by calculating the distance between the data to be classified and the existing training data. KNN is very simple and easy to understand but may face challenges in handling high-dimensional data [5].

The data used is the user review dataset of JKN Mobile taken from the Google Play Store. The analysis stage involves data preprocessing, such as text cleaning and extraction of relevant features. In the data retrieval process, the number of data ratings 1-5 is unbalanced, so the SMOTE step needs to be performed. SMOTE works by creating new synthetic data from the minority class through interpolation between existing data points. The synthetic data generated by SMOTE is in the feature space, not in the data space, because the scraping results are unbalanced [6]. Furthermore, the Naive Bayes, K-NN, and SVM methods are applied to the dataset. Evaluation is done using evaluation metrics such as accuracy, precision, recall, and F1-score.

The test results show that SVM has the highest accuracy of 85.71%, followed by Naive Bayes with 76.70% accuracy, and K-NN with 52.74% accuracy. Previous studies have also tested these three algorithms with similar results [7]. In another study, 5390 data were used for training and 599 data for testing. The results obtained from this research indicate that the accuracy of the K-Nearest Neighbor method is 54%, while the accuracy of the Support Vector Machine method with kernel is 79% [8]. There is also a similar study with accuracy results of 92.50% for Naive Bayes, 93.00% for SVM, and 95.00% for K-NN [9]. In this study, a single SVM algorithm was used with an accuracy of 82.20% [10].

METHOD

Research Methodology

Knowledge Discovery on Database (KDD) is considered a suitable method for this research, referring to the process of extracting useful knowledge from data in a database. This method consists of four main stages: data collection, data preprocessing, data mining, and evaluation [11].
Data Collection

Data collection was conducted by filtering user reviews of the JKN Mobile application from the Google Play Store using Google Colab. These reviews have been labeled as positive, negative, or neutral based on their content.

The number of negative reviews with a rating of 1-2 is 858, the number of neutral reviews with a rating of 3 is 80, and the number of positive reviews with a rating of 4-5 is 262. The total collected data is 1200. Since the data appears to be imbalanced, a SMOTE step is needed to balance the data used.

Preprocessing Data

Data preprocessing is an important step in preparing data before performing further analysis or processing. Its purpose is to clean, organize, and transform raw data into a more structured form that is easily understood and suitable for the analysis or modeling needs.

Several commonly used methods in data preprocessing are case folding, tokenizing, filtering, stopwords removal, stemming, and term weighting using TF-IDF [9].

Case Folding

Case Folding is the process of converting all text characters to either lowercase or uppercase to avoid irrelevant differences in writing.

Tokenizing

Tokenizing is the process of separating text into smaller units, such as words or tokens, to facilitate analysis.

Filtering

Filtering involves removing irrelevant or uninformative words in the analysis, such as common words (stopwords).

Stemming

Stemming transforms words into their base form by removing affixes or
endings, so that words with the same root can be considered as a single entity.

![Image 5. Stemming Results](image)

**Term Frequency-Inverse Document Frequency (TF-IDF)**

Term frequency (TF) adalah jumlah kata/term dalam suatu dokumen, sedangkan Inverse Document Frequency (IDF) adalah frekuensi kemunculan kata/term diseluruh dokumen[8]. TF-IDF bermanfaat dalam meningkatkan performance model, rumus TF-IDF 1.

$$\text{TF-IDF} = \frac{\text{word number } x}{\sum \text{word}} \log \frac{\text{document number } y}{\text{number/count}}$$  \hspace{1cm} (1)

Precision, recall, and F1-score formulas are used to measure the performance of the algorithm.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \times 100\%$$  \hspace{1cm} (2)

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100\%$$  \hspace{1cm} (3)

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100\%$$  \hspace{1cm} (4)

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \times 100\%$$  \hspace{1cm} (5)

**RESULT AND DISCUSSION**

**Splitting data**

The preprocessed data is then split into training data and test data. To determine the optimal result, the data is divided into 5 experiments: in experiment 1, the test data is 10% and the training data is 90%; in experiment 2, the test data is 20% and the training data is 80%; in experiment 3, the test data is 30% and the training data is 70%; in experiment 4, the test data is 40% and the training data is 60%; and finally, in experiment 5, the training data is 50% and the test data is 50%.

**Naive Bayes Classifier**

In each experiment, the division results in different accuracies, as shown in Table 1.
Table 1. Naïve Bayes Results

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-Score</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>88%</td>
<td>87%</td>
<td>86%</td>
<td>87%</td>
</tr>
<tr>
<td>2</td>
<td>86%</td>
<td>85%</td>
<td>85%</td>
<td>86%</td>
</tr>
<tr>
<td>3</td>
<td>86%</td>
<td>85%</td>
<td>85%</td>
<td>86%</td>
</tr>
<tr>
<td>4</td>
<td>86%</td>
<td>85%</td>
<td>85%</td>
<td>85%</td>
</tr>
<tr>
<td>5</td>
<td>83%</td>
<td>82%</td>
<td>81%</td>
<td>81%</td>
</tr>
</tbody>
</table>

From the table data, the most optimal result is obtained in the first experiment with an accuracy of 87%, precision of 88%, recall of 87%, and F1-score of 86%. The implementation of the most accurate classification resulted in the following Confusion Matrix.

![Confusion Matrix Naïve Bayes](image6)

In Image 6, the number of data with a positive value and correctly predicted as positive is 60, the number of data with a neutral value and correctly predicted as neutral is 93, the number of data with a negative value and correctly predicted as negative is 79. The number of positive data predicted as neutral is 17, the number of positive data predicted as negative is 6, the number of negative data predicted as positive is 2, and the number of negative data predicted as neutral is 4.

![Support Vector Machine Results](image7)

**Support Vector Machine (SVM)**

In each experiment, the division results in different accuracies, as shown in Table 2.

Table 2. Support Vector Machine Results

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-Score</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>97%</td>
<td>88%</td>
<td>92%</td>
<td>95%</td>
</tr>
<tr>
<td>2</td>
<td>95%</td>
<td>95%</td>
<td>94%</td>
<td>94%</td>
</tr>
<tr>
<td>3</td>
<td>94%</td>
<td>85%</td>
<td>89%</td>
<td>93%</td>
</tr>
<tr>
<td>4</td>
<td>90%</td>
<td>86%</td>
<td>88%</td>
<td>92%</td>
</tr>
<tr>
<td>5</td>
<td>90%</td>
<td>85%</td>
<td>88%</td>
<td>92%</td>
</tr>
</tbody>
</table>

Based on the table data, the most optimal result is obtained in the first experiment with an accuracy of 95%, precision of 97%, recall of 88%, and F1-score of 92%. The implementation of the most accurate classification resulted in the following Confusion Matrix.
In Image 8, the number of positive data correctly predicted as positive is 74, the number of neutral data correctly predicted as neutral is 93, the number of negative data correctly predicted as negative is 85. The number of positive data predicted as neutral and correctly classified is 6, and the number of positive data predicted as negative and correctly classified is 3.

In Image 9, the Support Vector Machine (SVM) results are as follows: accuracy = 83%, precision = 84%, recall = 78%, F1-score = 80%. The confusion matrix shows that the number of positive data correctly predicted as positive is 19, the number of neutral data correctly predicted as neutral is 93, the number of negative data correctly predicted as negative is 85. The number of positive data predicted as neutral and correctly classified is 48, and the number of positive data predicted as negative and correctly classified is 16.

### Table 3. K-Nearest Neighbor Results

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Results of K-Nearest Neighbor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
</tr>
<tr>
<td>1</td>
<td>83%</td>
</tr>
<tr>
<td>2</td>
<td>82%</td>
</tr>
<tr>
<td>3</td>
<td>81%</td>
</tr>
<tr>
<td>4</td>
<td>81%</td>
</tr>
<tr>
<td>5</td>
<td>80%</td>
</tr>
</tbody>
</table>

From the table data 3, the most optimal result is obtained in the first experiment with an accuracy of 75%, precision of 83%, recall of 73%, and F1-score of 68%. The implementation of the most accurate classification resulted in the following Confusion Matrix.

In Image 10, the number of positive data correctly predicted as positive is 19, the number of neutral data correctly predicted as neutral is 93, the number of negative data correctly predicted as negative is 85. The number of positive data predicted as neutral and correctly classified is 48, and the number of positive data predicted as negative and correctly classified is 16.
Overall Classification Results

From the five experiments conducted with three algorithms, the best result is obtained in the experiment with 10% test data and 90% training data. The overall classification results include precision, recall, F1-score, and accuracy.

Table 4. Overall Classification Results

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Overall Classification Results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
</tr>
<tr>
<td>NB</td>
<td>88%</td>
</tr>
<tr>
<td>SVM</td>
<td>97%</td>
</tr>
<tr>
<td>KNN</td>
<td>83%</td>
</tr>
</tbody>
</table>

From the comparison table 4, it can be observed that the SVM classification yields higher values compared to Naive Bayes and KNN. The SVM results show an accuracy of 95%, precision of 96%, recall of 95%, and F1-score of 95%. On the other hand, Naive Bayes achieves an accuracy of 87%, precision of 88%, recall of 87%, and F1-score of 86%, while KNN achieves an accuracy of 75%, precision of 83%, recall of 73%, and F1-score of 68%.

CONCLUSION

Based on the testing results of sentiment analysis on the JKN Mobile application using three algorithms, it can be concluded that out of the 1200 SMOTE-processed datasets with different proportions of training and test data, the experiment with 10% test data and 90% training data yields the most optimal results. The accuracies for Naive Bayes, Support Vector Machine (SVM), and K-Nearest Neighbor (KNN) in this experiment are 87%, 95%, and 75%, respectively. In the context of sentiment analysis for the JKN Mobile application, it is recommended to use the Support Vector Machine (SVM) algorithm due to its higher accuracy compared to Naive Bayes and K-Nearest Neighbor. Utilizing the Support Vector Machine can assist in predicting sentiments more accurately.

BIBLIOGRAPHY


