Hybrid Naive Bayes TF-IDF Algorithm and Lexicon Approach for Sentiment Analysis of Reviews

Ahmad Harits Ramadhani, Huzaifah Qahar Djauhari, Vincent Lius, Andi Nugroho

Abstract — Amidst the increasing reliance on social media for public expression, accurate sentiment analysis has become essential, notably in assessing application reviews. This study focuses on the need for precise sentiment classification, exemplified by the prevalence of negative feedback in certain app reviews. To address this, we propose a hybrid approach integrating the Naive Bayes algorithm with lexicon-based sentiment labeling and TF-IDF for the model training. Using a dataset of 5000 reviews, we explore Indonesian Lexicons, specifically InSet and SentiStrengthID, to label sentiments. Our primary objective is to classify reviews into positive and negative sentiments, providing valuable insights. Through evaluating the effectiveness of combining Naive Bayes with TF-IDF and lexicon-based methods, this study contributes to a deeper understanding of sentiment analysis in the context of application reviews.

Keywords — Sentiment Analysis, InSet Lexicon, SentiStrength Lexicon, Naive Bayes, Multinomial Naive Bayes, TF-IDF, Machine Learning.

I. INTRODUCTION

In recent years, sentiment analysis has emerged as a growing trend, particularly with the advent of social media and online platforms that facilitate individuals to share their opinions and views [1]. Alongside its popularity, the volume of generated data continues to escalate, necessitating more human resources, time, and effort to manage it [2][3][4]. However, to handle such vast and burgeoning data, effective techniques and approaches are required to extract meaningful insights from it.

Numerous studies have been conducted in the realm of sentiment analysis. In comparing machine learning algorithms for sentiment analysis on Twitter, Naive Bayes emerged as the most accurate algorithm with a score of 89%, followed by SVM with a score of 88%, and Random Forest with a score of 85% [5]. In the implementation of the Naive Bayes algorithm for sentiment analysis of Shopee reviews on the Google Play Store, the performance of Naive Bayes using the Hold-Out data splitting technique yielded a better accuracy of 83%, with precision of 83%, recall of 100%, and an f1-score of 91% compared to 10-fold cross-validation [6]. Furthermore, the accuracy of the Naive Bayes algorithm at 83.43% outperformed the Decision Tree algorithm accuracy at 82.91% and the Random Forest algorithm accuracy at 82.91% in sentiment analysis on Anti-LGBT cases on Twitter [7]. Moreover, the Naive Bayes algorithm combined with TF-IDF achieved an accuracy of 88.7%, higher than Word2vec with an accuracy of 83.3% [5], [8].

For sentiment analysis, a lexicon containing a list of words classified based on their sentiment, such as positive, negative, or neutral, is typically required. This lexicon is then used as a reference to classify the sentiment of the reviewed text. However, it is important to note that the accuracy of sentiment lexicon results may vary, especially if the lexicon is not regularly updated or does not encompass all possible words used in text reviews.

This paper is structured into five pivotal chapters. Chapter 1 offers an introduction, elucidating the research's background and objectives. Chapter 2, Methodology, provides a detailed exploration of the theories and techniques employed. In Chapter 3, Implementation, the practical application of these methodologies, from data preprocessing to model testing, is discussed. Following this, Chapter 4 presents the results of model evaluations, while Chapter 5, Conclusion, synthesizes findings and proposes future research directions, offering a cohesive conclusion to the study.

II. METHODOLOGY

Based on our research, lexicon methods such as InSet and SentiStrengthID, Naive Bayes classification algorithms, and TF-IDF techniques have demonstrated effectiveness in analyzing sentiments from app user reviews. Evaluation of the InSet lexicon for sentiment analysis in Indonesian text yielded a satisfactory accuracy of 65.78% in microblog [9]. Integrating the SentiStrengthID lexicon with Hybrid TF-IDF and Cosine Similarity achieved sentiment summarization from social media texts with 60% accuracy and 62% f-measure [10]. Moreover, the Naive Bayes algorithm effectively analyzed sentiment in Shopee app reviews on Google Play Store, achieving a satisfying 83% accuracy.

A. Word2Vec

Using artificial neural networks, Word2Vec aims to generate word vector representations that reflect the semantic meanings of words based on the context in which they appear [11]. The learning process involves two main models, namely Continuous Bag of Words (CBOW) and Skip-Gram, each focusing on predicting the target word from its surrounding context or vice versa.

B. Lexicon

A lexicon is a dictionary containing a list of words and

their sentiment value. Lexicons can be published in book form or accessed online. In sentiment analysis, the lexicon approach categorizes words based on the sentiments they convey, such as positive, negative, or neutral. In approaches that use a lexicon, a word can have different meanings depending on the domain being discussed. Therefore, a sentiment lexicon dictionary is required to categorize words according to the sentiments they convey. By using this dictionary, a sentence or phrase can be classified according to the polarity of the sentiments it contains [12].

Here are some data sources consisting of text collections in the Indonesian language that can be used in research utilizing the lexicon method:

1. Lexicon Inset

The InSet lexicon consists of 3,609 positive words and 6,609 negative words with weights ranging from -5 to +5.

2. Lexicon SentiStrengthID

The SentiStrengthID lexicon consists of 1,729 words with weights ranging from -5 to +5.

Both lexicons contain a list of words along with their corresponding weights that determine the polarity of a text. The polarity or sentiment score is calculated by summing the weights of the words that appear in a review [13]. The result of this calculation is then used as the sentiment class label, whether positive, negative, or neutral, for each aspect of the review.

The formulas used in the InSet and SentiStrengthID lexicon approaches are as follows [14] :

$$If\ sentiment\ score\ >\ 0\ then\ Sentimen\ Positif \quad (1)$$

$$If\ sentiment\ score\ <\ 0\ then\ Sentimen\ Negatif \quad (2)$$

$$If\ sentiment\ score\ =\ 0\ then\ Netral \quad (3)$$

Table 1 . Example of Lexicon InSet Calculation

Review (After Word Preprocessing) = ['bug', 'rilis', 'tanggung']				
Word	bug	rilis	tanggung	**Polarity**
Weight	-	3	-2	1

Table 2. Example of Lexicon SentiStrengthID Calculation

Review (After Preprocessing) = ['bug', 'rilis', 'tanggung']				
Word	bug	rilis	tanggung	**Polarity**
Weight	-	-	4	4

3. TF-IDF

Word Embedding is a feature learning technique that maps words into real-number vectors loure [15]. TF-IDF assigns weights to words based on their frequency within a document (TF) and across all documents (IDF). Words that are frequent in a specific document but rare in others receive higher weights, capturing their significance in that document's context. [16]. By using TF-IDF, feature selection in a corpus becomes more precise.

The formula used in TF-IDF is as follows:

$$TF(t,d) = \frac{N(t,d)}{T} \quad (4)$$

In equation (4), TF(t,d) represents the frequency of a word (t) appearing in a document (d), and T is the total number of words in a document.

$$DF(t) = log\ log \frac{N}{N(t)} \quad (5)$$

In equation (5), N represents the total number of documents, and N(t) represents the number of documents containing the word (t).

$$TF - IDF = TF * IDF \quad (6)$$

4. Naïve Bayes

Naive Bayes is a simple, popular, and powerful machine learning technique for classification [17]. Naive Bayes utilizes the Bayes theorem, employing a probabilistic approach [18]. Naive Bayes assumes that the presence of each condition or event is independent and not influenced or related to other conditions [19]. As the amount of data increases, Naive Bayes is suitable for use because it quickly classifies data and can be used directly [20].

The formula used in Naive Bayes is based on the Bayes theorem as follows [21].

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \quad (7)$$

P(A|B) = the probability of event A occurring given event B.
P(B|A) = the probability of event B occurring given event A.
P(A) = the probability of event A occurring.
P(B) = the probability of event B occurring.

Table 3. Naïve Bayes Algorithm

Algorithm
Input 1 : Set of encoded representations for each class (Y). **Input 2** : Training (X_{train}) with corresponding class labels (Y_{train}) **Input 3** : Test data (X_{test}) with actual class labels (Y_{test}) **Output** : Predicted class labels (Y_{pred}) for the test data (X_{test})
Calculate Instance Probabilities : Calculate the probability of each instances or word (ω) inside each feature X_i for each class Y_i, with formula as : $$P(\omega_i, Y_i) = \frac{N_{w_i Y_i} + \alpha}{N_{Y_i}}$$ with α = 1 as Laplace smoothing to avoid zero possibilities

Prior Probabilities of sentences belong to which class Y_i :

$$P(Y_i) = \frac{N_{Y_i}}{N}$$

Posterior Probabilities of entire sentence labeled to class Y_i :
product of individual feature probabilities multiplied with prior probabilities :

$$P(Y_i, X_j) = \frac{\prod_{i=1}^{L} P(\omega_i \mid Y_i) \cdot P(Y_i)}{N_{X_j}}$$

L = Total number of words (ω) in the sentence (X_j)
N_{X_j} is 1 considering each X_j is unique

Determine the Y_{pred} of X_j by comparing the highest $P(Y_i, X_j)$ amongst all Y_i

$$Y_{pred} = arg\ max_i\ P(Y_i, X_j)$$

5. Confusion Matrix

Confusion matrix is a method to measure the performance of a machine learning classification model by analyzing the predicted and actual results. It consists of 4 categories, namely True Positive (correct prediction according to actual result), True Negative (correct prediction opposite to actual result), False Positive (incorrect prediction according to actual result), and False Negative (incorrect prediction opposite to actual result)

Table 4. Confusion Matrix

Actual Class	Predicted Class	
	Positive	**Negative**
Positive	TP	FN
Negative	FP	TN

To evaluate the performance of the classification system, metrics such as accuracy, precision, recall, and f1-score are used, as defined by the following equations [22].

Accuracy measures how well the data is classified correctly. It is expressed by formula (8).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (8)$$

Precision refers to how well the data classified as a certain class truly belongs to that class and is measured for each class. Precision is measured using formula (9) below.

$$Precision = \frac{TP}{TP + FP} \quad (9)$$

Recall indicates how well the data that should be classified into a certain class is truly classified as that class. Recall is measured using formula (10) below.

$$Recall = \frac{TP}{TP + FN} \quad (10)$$

F1-Score (F-measure) is the average of precision and recall. F1-Score is measured using formula (11).

$$F1 - Score = 2 * \frac{presisi\ x\ recall}{presisi + recall} \quad (11)$$

6. Data Preprocessing

Data preprocessing involves several steps to make sentiment reviews easier to analyze. This stage typically begins by removing non-character data like digits, symbols, and punctuation, then converting the text to lowercase [23]. This preparation of data aims to facilitate processing and analysis, streamlining the subsequent stages of sentiment analysis.

6.1 Cleansing Data

In data cleansing, irrelevant elements like punctuation marks, links, icons, and hashtags are removed to enhance dataset quality. This improves analysis accuracy and efficiency

Table 5. Data Cleansing Example

Before	**After**
Banyak bug nya, kalo mau merilis jgn tnggung lah	Banyak bug nya kalo mau merilis jgn tnggung lah
The app is awesome!! Best ever!!!	The app is awesome. Best ever.

6.2 Case Folding

Case folding is a process that involves adjusting word forms by converting all letters to lowercase [24]. Only letters from 'a' to 'z' are accepted, and characters other than letters are removed. This is done so that same word on upper case and lower case is treated the same.

Table 6. Case Folding Example

Before	**After**
Banyak bug nya kalo mau merilis jgn tnggung lah	banyak bug nya kalo mau merilis jgn tnggung lah
The app is awesome. Best ever.	the app is awesome. best ever.

6.3 Tokenization

Tokenization is the process of breaking down a set of sentences into character pieces or words, commonly referred to as tokens [25]. This method utilizes spaces as the primary separator and considers punctuation marks and special characters, such as periods or commas, which can act as word separators or attach to specific words [26]. While effective in natural language processing as most human languages use spaces, word-based tokenization may encounter challenges in processing non-standard words or in the context of compound words or slang.

Table 7. Tokenization Example

Before	After
banyak bug nya kalo mau merilis jgn tnggung lah	['banyak', 'bug', 'nya', 'kalo', 'mau', 'merilis', 'jgn', 'tnggung', 'lah']
the app is awesome. best ever.	["the", "app", "is", "awesome"], ["best", "ever"]

6.4 Stopwords Removal

Stopwords removal aims to eliminate meaningless words, typically conjunctions or adverbs, from the parsed results of a text document [27]. This process involves comparing them with a stoplist containing frequently occurring words in documents, but not necessarily adding value to the information retrieval process. This step is a method for filtering out common words like "and", "or", and "etc." and similar ones that are not needed in data processing [28].

Table 8. Stopwords Removal Example

Before	After
['banyak', 'bug', 'nya', 'kalo', 'mau', 'merilis', 'jgn', 'tnggung']	['bug', 'merilis', 'tnggung']

From the table above, it can be seen that several words that do not have significant meaning or importance are removed in this process, such as the words 'banyak', 'nya', 'kalo', 'mau', 'jgn', which are stopwords in the Indonesian language

6.5 Stemming

Stemming is the process of removing affixes from each word, converting them into their base form [29]. This is done so that words with or without affixes are considered the same in analysis.

Table 9. Stemming Example

Before	After
['bug', 'merilis', 'tanggung']	['bug', 'rilis', 'tanggung']
[‘app,’ ‘running’]	[‘app’, ‘run’]

6.6 Normalization

Normalization in text analysis aimed at standardizing terms or words that essentially have the same meaning but vary in spelling or structure. This often occurs due to writing errors, spelling variations, word abbreviations, or the use of terms in informal or slang language [30]. Normalization is performed to avoid unnecessary duplication and reducing the dimensionality of word counts

Table 10. Normalization Example

Before	After
['bug', 'merilis', 'tnggung']	['bug', 'merilis', 'tanggung']
[‘this’, ‘app’, ‘awsum‘]	[‘this’, ‘app’, ‘awesome‘]

III. IMPLEMENTATION

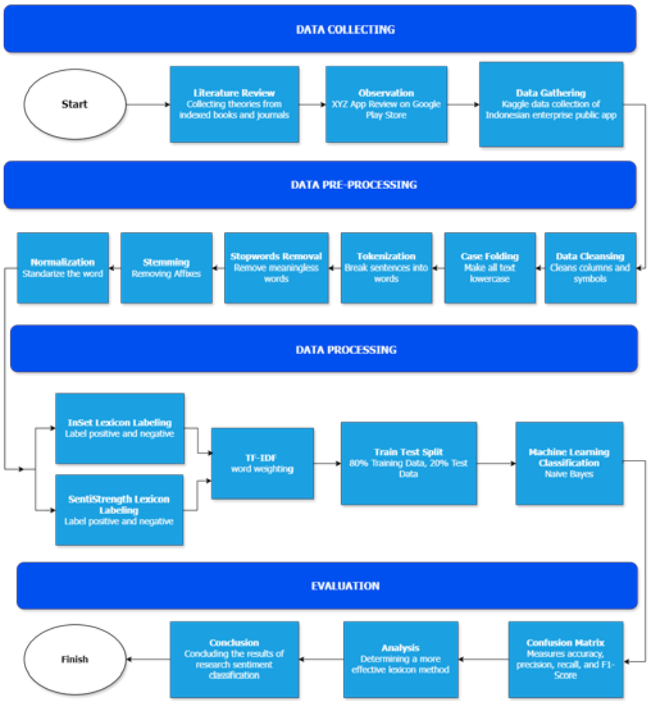

Figure 1. Research Flow Diagram

In the Dataset Overview phase, where its structure and content are explored to guide subsequent steps. Data Preprocessing follows, involving cleaning, normalization, and transformation to ensure data quality and suitability for analysis.

Train Data Labeling with Lexicon assigns labels or categories using custom rules, on this case each sentiment score of the word are added up. If the total score is < 0 then it will be labeled negative and vice versa.

Feature Engineering with TF-IDF converts text data into numerical features, capturing word importance using techniques like TF-IDF.

Finally, Model Training with multinomial naive bayes models, trained on preprocessed data and engineered features data to uncover patterns and relationships, facilitating prediction or classification tasks.

1. Dataset Overview

For our research, we acquired a dataset from Kaggle comprising 5000 Indonesian app reviews gathered from diverse origins. These reviews present firsthand experiences with enterprise-level public applications, offering valuable insights into user sentiments and perspectives. Each review offers comprehensive feedback, providing a holistic view of user interactions with the applications under examination. Through analysis of this Kaggle dataset, our objective is to identify patterns, trends, and sentiments to inform decision-making and enhance user experience and functionality of enterprise applications in the Indonesian market.

Table 11. Example of Dataset

Username	Content	Score
Indra Setiyadi	Dapat driver jemput pada jauh2 banget bikin telat kerja	1
IsmiHadianti	Lintah memang Pinjaman 22k telat sehari karna Apk harus di upgrade Denda 50k	1
Melova Cantik	Habis isi gopay 300000 baru dipakai dikit sisa masih banyak apk gojek g bisa dibuka sdh 2 bl lalu	1
arif Abdul hidayat	Sangat membantu	5

2. Data Preprocessing

In this section, we describe the steps undertaken to preprocess the textual data we obtained. The preprocessing steps are designed to clean and standardize the textual content, making it amenable to further analysis and modeling.

Table 12. Word Cleansing and Case Folding

Before	After
Aplikasi ini luar biasa! Sangat ramah pengguna dan membantu saya tetap teratur.	aplikasi ini luar biasa sangat ramah pengguna dan membantu saya tetap teratur

Special characters such as emoticons, non-ASCII characters, mentions, links, and hashtags are removed from the text. This step ensures that only meaningful content remains for analysis. Numeric characters, punctuations, leading and trailing white spaces are also removed from the text since they often do not contribute to the semantic meaning of the content. All words are lowercased to prevent the program on reading it as different character.

Table 13. Tokenization and Stopword Removal

Before	After
aplikasi ini keren dan sangat ramah pengguna dan membantu saya tetap teratur	["aplikasi", "keren", "ramah", "pengguna", "membantu", "tetap", "teratur"]

Tokenization is done to ensure that each word will become separate element on sentiment analysis process. While stopword removal is done to remove meaningless word.

Table 14. Stemming and Normalization

Before	After
["aplikasi", "keren", "ramah", "pengguna", "membantu", "tetap", "teratur"]	["aplikasi", "keren", "ramah", "guna", "bantu", "tetap", "atur"]

Stemming is applied to reduce inflected or derived words to their base or root form. This process aims to normalize variations of words and improve text coherence. Text normalization involves replacing slang or informal terms with their corresponding formal equivalents. This step enhances the consistency and interpretability of the textual data.

3. Train Data Labeling with Lexicon

To incorporate this lexicon into our model training process, we perform lexicon labeling on the training data. This involves annotating each instance in the training dataset with sentiment scores derived from the SentiStrength-Inset lexicon. By leveraging the sentiment scores provided by the lexicon, we enrich the training data with additional features that capture the underlying sentiment of the text. While also makes the process of labeling training data that is commonly done manually to be automated and more efficient.

Sentences that do not contain sentiment-related words or are not present in the lexicon are excluded from the training data to ensure that only informative instances are retained.

This lexicon is represented as L. The sentiment score S for the text is calculated by summing the sentiment scores of individual words present in the lexicon. Mathematically, it can be expressed as:

$$S = \sum_{i=1}^{n} score_i \tag{12}$$

Where n denotes the total number of words in the text, score i represents the sentiment score of the i th word, and i ranges from 1 to n.The polarity P of the text is determined based on the sentiment score
S as follows

$$\mathrm{P} = \begin{cases} 'positive' & if\ S > 0 \\ 'no-sentiment' & if\ S = 0 \\ 'negative' & if\ S < 0 \end{cases} \tag{13}$$

Where P represents the polarity of the text, and the conditions are evaluated using standard comparison operators.

```
polarity
positive     2072
negative     2027
Name: count, dtype: int64
```

Figure 2. Result of InSet Labelling

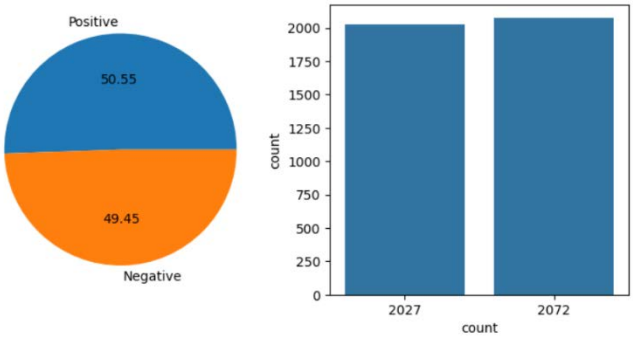

Figure 3. Visualization of InSet Labelling

In **Figure 2.** we can observe the results of the lexicon inset labeling, which yielded a positive polarity of 2072 and a negative polarity of 2027 out of 5000 dataset. There are some reviews that do not have sentiment or can be called neutral, amounting to 901. We decided to remove the neutral labels. Then, in **Figure 3.** the results are visualized using pie charts and bar diagrams, indicating that the percentage of positive scores is 50.55%, while negative scores account for 49.45%.

```
polarity
positive     2098
negative      960
Name: count, dtype: int64
```

Figure 4. Result of SentiStrength Labelling

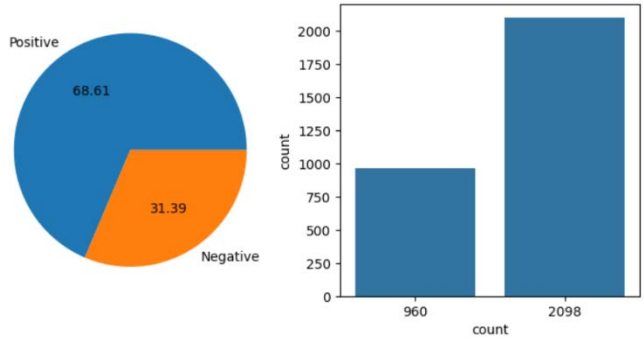

Figure 5. Visualization of SentiStrength Labelling

In **Figure 4.** the results of lexicon SentiStrengthID labeling show polarities with 2098 positive, 960 negative, and 1942 neutral values out of 5000 dataset.. Then, in **Figure 5.** these results are visualized using pie charts and bar diagrams, indicating that the percentage of positive scores is 68.61%, while positive scores account for 31.39%.

Given the potential occurrence of sentences devoid of discernible emotion, resulting in a polarity score of 0, a practical approach is introduced. Here, sentences lacking emotional sentiment are labeled as 'no-sentiment' and subsequently excluded from both the training and testing datasets. This strategic exclusion aims to refine sentiment analysis models by focusing solely on text segments rich in emotional content, thereby enhancing training efficacy and overall classification system performance.

4. Feature Engineering with TF-IDF

For our feature engineering process, we initialize a TfidfVectorizer object *tfidf* to facilitate the TF-IDF transformation on the labeled text data. This object will be utilized to learn the TF and IDF (Inverse Document Frequency) parameters necessary for the transformation.

Once the *tfidf* object is trained, we partition the data into training and testing sets using the train_test_split function from the sklearn.model_selection module. The resulting training set, denoted as *train_data*, and *test_data*.

With the data split, we proceed to transform the stemmed text data into TF-IDF vectors for both the training and testing sets using the fitted *tfidf* object. This transformation yields numerical representations X_{train} and X_{test} suitable for input into machine learning models.

Label the testing data based on the score (polarity), where a score of 3 or higher is labeled as positive (1), and scores below 3 are labeled as negative (0). This can be represented using an indicator function where:

$$L(s) = \begin{cases} 1 \ if \ s \geq 3 \\ 0 \ if \ s < 3 \end{cases} \quad (14)$$

where $score_i$ represents the sentiment score of the ith review in the testing set, and $Y_{test}(i)$ denotes the label for the ith review.

The transformed data matrices (X_{train}, X_{test}) along with their respective labels (Y_{train}, Y_{test}) are returned as output. These data sets are essential for further analysis and model training, providing the foundation for sentiment analysis and predictive modeling tasks.

5. Model Training

Initialize a Multinomial Naive Bayes classifier and train it using the training data or X_{train}. The Multinomial Naive Bayes classifier designed to take X_{train} as feature in a form of categorical counts, representing TF-IDF score in text data, and it uses these score to estimate the probabilities of different classes in Y_{train}.

Then, predict the labels for the testing data or X_{test} to obtain Y_{pred} using **classifier.predict(X_{test})**. For evaluating the model's performance, function to calculate the precision, recall, accuracy, and F1-score will be initialized.

$$\begin{aligned} &\text{precision} = \text{calculate\_precision}\ (Y_{test}, Y_{pred}) \\ &\text{recall} = \text{calculate\_recall}\ (Y_{test}, Y_{pred}) \\ &\text{accuracy} = \text{calculate\_accuracy}\ (Y_{test}, Y_{pred}) \\ &\text{f1\_score=calculate\_f1\_score}\ (Y_{test}, Y_{pred}) \end{aligned} \quad (15)$$

Predictions that are made on the testing data are compared with label of manually sentiment scored test data. Model's

performance is categorized with various metrics, including precision, recall, accuracy, and F1-score. These metrics provide valuable insights into the classifier's ability to accurately classify text data and serve as benchmarks for assessing its efficacy.

```
Accuracy Score - 0.6865853658536586
Precision Score - 0.7901866325293087
Recall Score - 0.6865853658536586
[[229  28]
 [229 334]]
F1 Score - 0.6965866750256994
```

Figure 6. Result of InSet Lexicon using Naïve Bayes

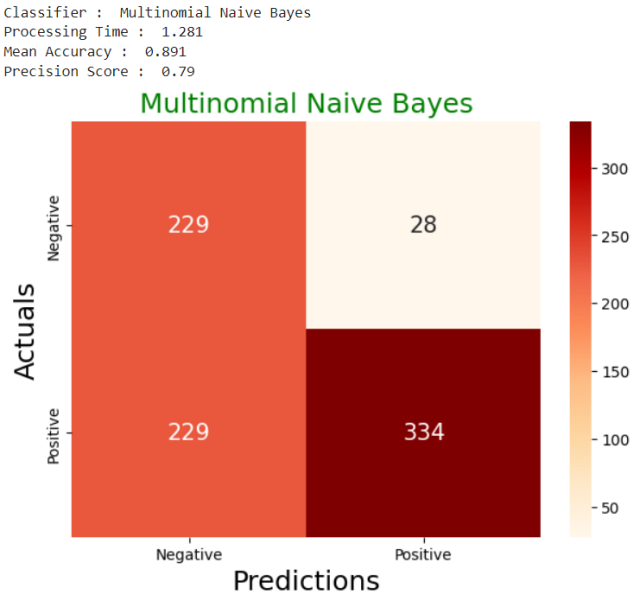

Figure 7. Visualization of InSet Lexicon's Result using Naïve Bayes

The evaluation of the model's performance using Naive Bayes method with lexicon InSet yielded an accuracy score of 0.686, reflecting the overall correctness in classification. A precision score of 0.790 indicates the proportion of correctly identified positive instances out of all instances classified as positive. Similarly, the recall score, matching the accuracy at 0.686, demonstrates the model's ability to correctly identify positive instances from the total number of actual positive instances. Finally, the F1-score is 0.696 reflecting the overall model performance. Specifically, there were 334 true positives and 229 true negatives, indicating instances correctly classified as positive and negative, respectively. However, there were 28 false positives and 229 false negatives, indicating instances incorrectly classified as positive and negative, respectively. These evaluation results offer a comprehensive understanding of the model's performance, highlighting both its strengths and areas for improvement.

```
Accuracy Score - 0.8464052287581699
Precision Score - 0.8444547335197633
Recall Score - 0.8464052287581699
[[134  52]
 [ 42 384]]
F1 Score - 0.845174674964139
```

Figure 8. Result of SentisStrength Lexicon using Naïve Bayes

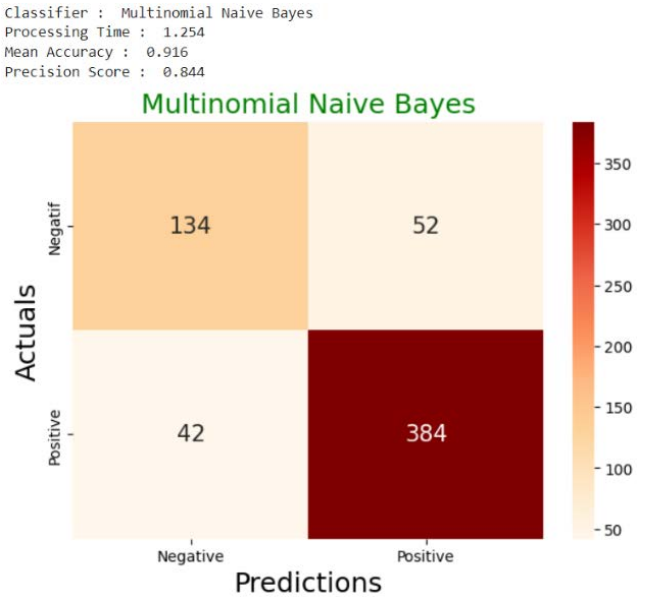

Figure 9. Visualization of SentiStrength Lexicon's Result using Naïve Bayes

The evaluation of the Naive Bayes model using lexicon SentiStrength yielded promising results. The accuracy score of 0.846 indicates the overall correctness in classification, with a precision score matching at 0.844, signifying the proportion of correctly identified positive instances out of all instances classified as positive. Additionally, the recall score also reached 0.846, demonstrating the model's effectiveness in identifying positive instances from the total number of actual positive instances. Finally, the F1-score is 0.845 reflecting the overall model performance. Specific to the classifications, there were 384 true positives and 134 true negatives, indicating instances correctly classified as positive and negative, respectively. However, there were 52 false positives and 42 false negatives, suggesting instances incorrectly classified as positive and negative, respectively. These results showcase the model's strong performance, with potential areas for further refinement.

IV. RESULT

Table 15. Result of InSet and SentriStrength Lexicon using Naïve Bayes

Lexicon Inset

Inset			
Accuracy	Precision	Recall	F1Score
68,6%	79%	68,6%	69,6%

Lexicon SentiStrength

SentiStrength			
Accuracy	Precision	Recall	F1Score
84,6%	84,4%	84,6%	84,5%

The results underscore the substantial impact of lexicon choice on sentiment analysis performance, with Lexicon SentiStrength outperforming Lexicon InSet across all evaluated metrics. With this result, we can safely conclude that Lexicon SentiStrength performs better on enhancing model training process on sentiment analysis.

V. CONCLUSION

In this study, we tackled the pressing need for accurate sentiment analysis in evaluating application reviews, employing a hybrid approach that combines the Naive Bayes algorithm with lexicon-based sentiment labeling and TF-IDF for model training. Through meticulous preprocessing and labeling of training data with Indonesian lexicons, namely InSet and SentiStrengthID, and subsequent word weighting using TF-IDF, our goal was to effectively categorize reviews into positive and negative sentiments. Our investigation revealed the critical influence of lexicon selection on sentiment analysis performance, with Lexicon SentiStrengthID demonstrating superior efficacy over Lexicon InSet across all evaluation metrics.

This study contributes valuable insights into sentiment analysis within the realm of application reviews, emphasizing the importance of leveraging advanced methodologies to enhance classification accuracy. Moving forward, future research endeavors could focus on refining lexicon-based methods and exploring alternative algorithms to further optimize sentiment analysis performance. By advancing our understanding of sentiment analysis techniques, we aim to facilitate informed decision-making and elevate user experience in application development through data-driven insights.

REFERENCES

[1] H. Barakat, R. Yeniterzi, and L. Martín-Domingo, "Applying deep learning models to twitter data to detect airport service quality," *J Air Transp Manag*, vol. 91, Mar. 2021, doi: 10.1016/j.jairtraman.2020.102003.

[2] Q. A. Xu, V. Chang, and C. Jayne, "A systematic review of social media-based sentiment analysis: Emerging trends and challenges," *Decision Analytics Journal*, vol. 3, p. 100073, Jun. 2022, doi: 10.1016/j.dajour.2022.100073.

[3] M. V. Mäntylä, D. Graziotin, and M. Kuutila, "The evolution of sentiment analysis—A review of research topics, venues, and top cited papers," *Computer Science Review*, vol. 27. Elsevier Ireland Ltd, pp. 16–32, 2018. doi: 10.1016/j.cosrev.2017.10.002.

[4] S. Hassan, C. Tantithamthavorn, C. P. Bezemer, and A. E. Hassan, "Studying the dialogue between users and developers of free apps in the Google Play Store," *Empir Softw Eng*, vol. 23, no. 3, pp. 1275–1312, Jun. 2018, doi: 10.1007/s10664-017-9538-9.

[5] A. Nayak and S. Natarajan, "Comparative study of Naïve Bayes, Support Vector Machine and Random Forest Classifiers in Sentiment Analysis of Twitter feeds," *International Journal of Advanced Studies in Computer Science and Engineering*, vol. 5, no. 1, 2016.

[6] N. C. Agustina, D. Herlina Citra, W. Purnama, C. Nisa, and A. Rozi Kurnia, "The Implementation of Naïve Bayes Algorithm for Sentiment Analysis of Shopee Reviews on Google Play Store Implementasi Algoritma Naive Bayes untuk Analisis Sentimen Ulasan Shopee pada Google Play Store," *MALCOM: Indonesian Journal of Machine Learning and Computer Science* , vol. 2, pp. 47–54, 2022.

[7] V. A. Fitri, R. Andreswari, and M. A. Hasibuan, "Sentiment analysis of social media Twitter with case of Anti-LGBT campaign in Indonesia using Naïve Bayes, decision tree, and random forest algorithm," in *Procedia Computer Science*, Elsevier B.V., 2019, pp. 765–772. doi: 10.1016/j.procs.2019.11.181.

[8] I. R. Hendrawan, E. Utami, and A. D. Hartanto, "Comparison of Naïve Bayes Algorithm and XGBoost on Local Product Review Text Classification," *Edumatic: Jurnal Pendidikan Informatika*, vol. 6, no. 1, pp. 143–149, Jun. 2022, doi: 10.29408/edumatic.v6i1.5613.

[9] K. Fajri and R. Gemala Y, "InSet Lexicon: Evaluation of a Word List for Indonesian Sentiment Analysis in Microblogs," in *International Conference on Asian Language Processing (IALP)*, 2017, pp. 391–394.

[10] D. Haryalesmana Wahid, "Peringkasan Sentimen Esktraktif di Twitter Menggunakan Hybrid TF-IDF dan Cosine Similarity," *IJCCS*, vol. 10, no. 2, pp. 207–218, 2016.

[11] Naufal Adi Nugroho and Erwin Budi Setiawan, "Implementation Word2Vec for Feature Expansion in Twitter Sentiment Analysis," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 5, pp. 837–842, Oct. 2021, doi: 10.29207/resti.v5i5.3325.

[12] N. R. Bhowmik, M. Arifuzzaman, and M. R. H. Mondal, "Sentiment analysis on Bangla text using extended lexicon dictionary and deep learning algorithms," *Array*, vol. 13, Mar. 2022, doi: 10.1016/j.array.2021.100123.

[13] S. Anggina, N. Y. Setiawan, and F. A. Bachtiar, "Analisis Ulasan Pelanggan Menggunakan Multinomial Naïve Bayes Classifier dengan Lexicon-Based dan TF-IDF Pada Formaggio Coffee and Resto," *@is The Best Accounting Information Systems and Information Technology Business Enterprise this is link for OJS us*, vol. 7, no. 1, pp. 76–90, Sep. 2022, doi: 10.34010/aisthebest.v7i1.7072.

[14] I. F Fanhar Nur, A. Herdiani, and W. Astuti, "Analisis Sentimen Berbasis Leksikon InSet Terhadap Partai Politik Peserta Pemilu 2019 Pada Media Sosial Twitter," in *e-Proceeding of Engineering : Vol.6, No.3*, 2019, pp. 10397–10407.

[15] M. L. Loureiro, M. Alló, and P. Coello, "Hot in Twitter: Assessing the emotional impacts of wildfires with sentiment analysis," *Ecological Economics*, vol. 200, Oct. 2022, doi: 10.1016/j.ecolecon.2022.107502.

[16] A. S. Neogi, K. A. Garg, R. K. Mishra, and Y. K. Dwivedi, "Sentiment analysis and classification of Indian farmers' protest using twitter data," *International Journal of Information Management Data Insights*, vol. 1, no. 2, Nov. 2021, doi: 10.1016/j.jjimei.2021.100019.

[17] N. A. Mansour, A. I. Saleh, M. Badawy, and H. A. Ali, "Accurate detection of Covid-19 patients based on Feature Correlated Naïve Bayes (FCNB) classification strategy," *J Ambient Intell Humaniz Comput*, vol. 13, no. 1, pp. 41–73, Jan. 2022, doi: 10.1007/s12652-020-02883-2.

[18] T. Olsson, M. Ericsson, and A. Wingkvist, "To automatically map source code entities to architectural modules with Naive Bayes," *Journal of Systems and Software*, vol. 183, Jan. 2022, doi: 10.1016/j.jss.2021.111095.

[19] H. Noviyarto, "Implementation Regression and Naïve Bayes to Predict and Classify Data Asset at Educational Institutions," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, vol. 2, no. 12, pp. 22–25, 2020.

[20] Hubert, P. Phoenix, R. Sudaryono, and D. Suhartono, "Classifying Promotion Images Using Optical Character Recognition and Naïve Bayes Classifier," in *Procedia Computer Science*, Elsevier B.V., 2021, pp. 498–506. doi: 10.1016/j.procs.2021.01.033.

[21] M. Artur, "Review the performance of the Bernoulli Naïve Bayes Classifier in Intrusion Detection Systems using Recursive Feature Elimination with Cross-validated selection of the best number of features," in *Procedia Computer Science*, Elsevier B.V., Jul. 2021, pp. 564–570. doi: 10.1016/j.procs.2021.06.066.

[22] D. Gamal, M. Alfonse, E. S. M. El-Horbaty, and A. B. M. Salem, "Implementation of Machine Learning Algorithms in Arabic Sentiment Analysis Using N-Gram Features," in *Procedia Computer Science*, Elsevier B.V., 2018, pp. 332–340. doi: 10.1016/j.procs.2019.06.048.

[23] V. Umarani, A. Julian, and J. Deepa, "Sentiment Analysis using various Machine Learning and Deep Learning Techniques," *Journal of the Nigerian Society of Physical Sciences*, vol. 3, no. 4, pp. 385–394, Nov. 2021, doi: 10.46481/jnsps.2021.308.

[24] A. G. Gozal, H. Pranoto, and M. F. Hasani, “Sentiment analysis of the Indonesian community toward face-to-face learning during the Covid-19 pandemic,” in *Procedia Computer Science*, Elsevier B.V., 2023, pp. 398–405. doi: 10.1016/j.procs.2023.10.539.

[25] R. Ahuja, A. Chug, S. Kohli, S. Gupta, and P. Ahuja, “The impact of features extraction on the sentiment analysis,” in *Procedia Computer Science*, Elsevier B.V., 2019, pp. 341–348. doi: 10.1016/j.procs.2019.05.008.

[26] S. Tabinda Kokab, S. Asghar, and S. Naz, “Transformer-based deep learning models for the sentiment analysis of social media data,” *Array*, vol. 14, Jul. 2022, doi: 10.1016/j.array.2022.100157.

[27] K. Madatov, S. Beckhanov, and J. Vi˘ci˘c, “Dataset of stopwords extracted from Uzbek texts,” *Data Brief*, vol. 43, 2022, doi: 10.5281/zenodo.6319953.

[28] R. Rani and D. K. Lobiyal, “Performance evaluation of text-mining models with Hindi stopwords lists,” *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 6, pp. 2771–2786, Jun. 2022, doi: 10.1016/j.jksuci.2020.03.003.

[29] T. H. J. Hidayat, Y. Ruldeviyani, A. R. Aditama, G. R. Madya, A. W. Nugraha, and M. W. Adisaputra, “Sentiment analysis of twitter data related to Rinca Island development using Doc2Vec and SVM and logistic regression as classifier,” in *Procedia Computer Science*, Elsevier B.V., 2021, pp. 660–667. doi: 10.1016/j.procs.2021.12.187.

[30] K. Ahmed *et al.*, “Breaking down linguistic complexities: A structured approach to aspect-based sentiment analysis,” *Journal of King Saud University - Computer and Information Sciences*, vol. 35, no. 8, Sep. 2023, doi: 10.1016/j.jksuci.2023.101651.

Ahmad Harits Ramadhani - Computer Science Department, Mercu Buana University, Jakarta, Indonesia Email: 41820010051@student.mercubuana.ac.id

Huzaifah Qahar Djauhari - Computer Science Department, Mercu Buana University, Jakarta, Indonesia Email: 41820010070@student.mercubuana.ac.id

Vincent Lius - Computer Science Department, Mercu Buana University, Jakarta, Indonesia Email: 41820010037@student.mercubuana.ac.id

Andi Nugroho - Phd Student in Computer Sciences – Computer Science Department, Mercu Buana University, Jakarta, Indonesia Email : andi.nugroho@mercubuana.ac.id Scopus Author ID : 57208427717 ORCID : orcidID= https://orcid.org/0000-0002-1713 - 035X.