

## Voting classifier in pain points identification

Yusup Miftahuddin, Muhammad Alif Firdaus

Department of Informatics, Faculty of Industrial Technology, National Institute of Technology Bandung, Bandung, Indonesia

### Article Info

#### Article history:

Received Jun 11, 2024

Revised Jul 21, 2025

Accepted Aug 6, 2025

#### Keywords:

Accuracy

F1-score

Identification

Pain point

Voting classifier

### ABSTRACT

A successful app understands and addresses the needs of its users. Pain points-specific difficulties and frustrations that users experience while using an application are crucial for understanding user expectations and improving user experience. Google Play Store reviews can be a valuable source for identifying these pain points, but this raw data requires processing to be useful for developers. This study develops a model to automatically classify reviews as either containing pain points or not. We chose the voting classifier as our primary algorithm because of its proven ability to produce models with high accuracy through combining the strengths of multiple classifiers. After evaluating 5 different classifier methods, our research shows that the optimal model combines XGradient boosting, multinomial naïve Bayes, and logistic regression with each contributing unique strengths in text classification. This combination achieves 90% accuracy and a 90% F1-Score, outperforming previous studies that used neural networks (which achieved 80% accuracy). The model successfully identifies user frustrations from app reviews, providing developers with actionable insights to improve their applications.

This is an open access article under the [CC BY-SA](#) license.



### Corresponding Author:

Yusup Miftahuddin

Department of Informatics, Faculty of Industrial Technology, National Institute of Technology Bandung  
Bandung, Indonesia

Email: yusufm@itenas.ac.id

## 1. INTRODUCTION

Pain points are specific obstacles or difficulties that users experience while using an application. These difficulties occur when an application fails to meet user expectations, leading to frustration and disappointment [1]. For developers of widely-used applications, identifying and addressing these pain points is essential to enhance user comfort and satisfaction. From a marketing perspective, understanding pain points can drive business development and subsequently generate profits [1].

Venkatakrishnan *et al.* [2] noted that while developers are generally aware of customer interests and have a clear understanding of their target audience, they are limited in their ability to improve their applications without appropriate feedback. Customer feedback and ratings serve as key metrics for evaluating performance and providing recommendations to enhance an app's functionality. This feedback is crucial for developers who seek to improve their applications [2].

In today's data-rich environment, the Google Play Store represents a valuable source of pain point data through user reviews. As Karim *et al.* [3] stated in their research on classifying Google Play Store application reviews, this platform has become a leading channel for downloading and uploading Android applications. Users frequently express their experiences—both positive and negative—through Google Play Store reviews [3]. These reviews represent direct opinions from users and significantly influence other potential users' decisions when selecting applications [4]. However, not all reviews contain pain points, necessitating a ML model that can effectively classify reviews as either containing pain points or not.

For the feature engineering process in this study, we employ term frequency-inverse document frequency (TF-IDF). This technique is particularly appropriate for text classification tasks as it effectively weighs the importance of words in a document relative to a collection of documents. TF-IDF has been proven to provide good results in several ML identification studies [5], [6], as it helps remove noisy and less relevant data while focusing on the most meaningful terms for classification.

We train our dataset using the voting classifier algorithm with a combination of three classification algorithms. These three algorithms were selected from a comparison of five classification algorithms based on their high cross-validation values in similar studies [7]. The selection criteria focused on determining which three classification algorithms would yield the highest accuracy and F1-score. We measure model performance using accuracy and F1-score metrics, as these are appropriate for our balanced dataset of pain points and non-pain points. In contrast, other studies have used area under the curve (AUC) as a performance parameter when dealing with unbalanced data [8].

Our dataset consists of reviews for the Shopee application collected from the Google Play Store. These reviews were manually labeled as either "Pain Point" or "Non-Pain Point" with validation from a UX Researcher. Through our research approach and carefully curated dataset, we aim to create a model with high accuracy and F1-score that can be utilized by developers-especially Shopee application developers-to distinguish between pain points and non-pain points in their application reviews. As Latif *et al.* [9] suggested in their research on data scraping from the Google Play Store, such analysis can significantly help developers improve the performance and efficiency of their applications.

## 2. RELATED WORK

Previous research on pain point identification has achieved varying levels of success. Salminen *et al.* [1] conducted a study to identify pain points from Twitter user tweets about 20 world-famous brands across 5 different industries. They compared several ML algorithms and found that neural networks produced the highest accuracy (80%) and F1-Score for pain point detection. Our study builds upon this foundation while exploring a different approach to improve these results.

In this paper, we employ ensemble learning-a ML technique that combines several classifiers to produce models with better identification performance than using individual classifiers alone [10]. The voting classifier, a specific ensemble learning algorithm, has demonstrated effectiveness in various identification studies. Mahabub, in their research on fake news detection, compared several classifiers including multilayer perceptron, X-gradient boosting, random forest (RF), multinomial naive Bayes (MNB), and logistic regression (LR) using cross-validation. They selected the three classifiers with the highest cross-validation values to create a voting classifier combination, which achieved an impressive 94.5% accuracy with a 95% F1-score [7].

Similarly, Elsaeed *et al.* [11] conducted research on detecting fake news on social media using voting classifiers with three different datasets: Fake-or-Real-News, Media-Eval, and ISOT. After dividing the datasets using K-fold cross-validation and employing TF-IDF and DOC2VEC for feature extraction (with feature selection via chi-square and ANOVA procedures), they achieved high model accuracy: 94.5% for the Fake-or-Real-News dataset, 91.2% for the Media-Eval dataset, and 100% for the ISOT dataset [11].

Salamai *et al.* [8] investigated dynamic voting classifiers for risk identification in supply chain 4.0. After building models using single classifiers such as support vector machine, neural network, and k-nearest neighbors, they found that the highest AUC value was only 0.823 (using the RF method). However, when they implemented the Sin Cosine Dynamic Group (SCDG)-based voting classifier, bagging, and majority-based ensemble learning methods, the highest AUC value improved dramatically to 0.989 using the SCDG-based voting classifier [8]. This demonstrates that ensemble learning techniques can produce significantly higher performance metrics than individual classification algorithms.

Various implementations of voting classifiers have been successfully applied across different domains. Chandra *et al.* [12] created a majority voting-based classifier ensemble consisting of seven benchmark supervised models for COVID-19 diagnosis. This ensemble approach reduced false diagnoses by aggregating predictions from multiple models, enhancing the system's robustness and accuracy. The majority voting technique particularly improved classification reliability in scenarios where misclassification could have serious consequences due to COVID-19's highly contagious nature [12].

Rai *et al.* [13] proposed a soft voting ensemble classifier for predicting respiratory failure in COVID-19 patients. Their approach combined the probabilities predicted by individual classifiers-including RF, XGBoost, gradient boosting classifier, and extra tree classifier-and selected the class with the highest probability. This soft voting ensemble improved the overall predictive performance and accuracy of their system [13]. In gas sensor applications, Alimisis *et al.* [14] and Kibria *et al.* [15] developed weighted voting

classifiers that assigned different weights to individual classifiers based on similarity indices of signals. This weighted approach enhanced classification accuracy by prioritizing more reliable classifiers in the ensemble.

For hate speech detection, Balouchzahi *et al.* [16] implemented a hybrid ensemble of classifiers with a voting scheme. By integrating multiple classifiers and utilizing a voting scheme, they improved hate speech detection performance by leveraging the complementary strengths of different classifiers. Faisal *et al.* [17] introduced an adaptive voting classifier for predicting movie quality. Their adaptive approach adjusted weights assigned to each individual classifier based on performance during training, allowing the ensemble to dynamically prioritize better-performing classifiers and improve movie quality prediction.

In network security applications, Khafaga *et al.* [18] demonstrated that integrating a voting classifier in network intrusion detection systems significantly enhanced attack detection accuracy and efficiency. Their approach showed superior performance compared to existing methods when tested with real-world IoT network datasets. The voting classifier, combined with metaheuristic optimization algorithms, improved attack detection precision and exhibited robustness in handling diverse intrusion scenarios, effectively leveraging the collective predictions of multiple classifiers to make more accurate decisions [18]. These studies consistently demonstrate that ensemble-based voting classifiers outperform individual classifiers across various domains, providing a strong rationale for our approach to pain point identification in app reviews.

### 3. METHOD

This research focuses on maximizing accuracy and F1-score through identification modeling using voting classifiers. We collected 1,000 reviews from the Google Play Store, comprising 500 pain point reviews and 500 non-pain point reviews. The attributes extracted include identity (pain point or non-pain point), username, score, timestamp, and content (review text).

As illustrated in Figure 1, our modeling approach using the voting classifier begins with data collection and labeling to create our dataset. Next, we pre-process the dataset-handling missing values, tokenizing text, and removing stopwords. We then apply feature engineering using TF-IDF, followed by dataset splitting. Finally, we train our model, validate it, and evaluate its performance to assess the effectiveness of our approach.

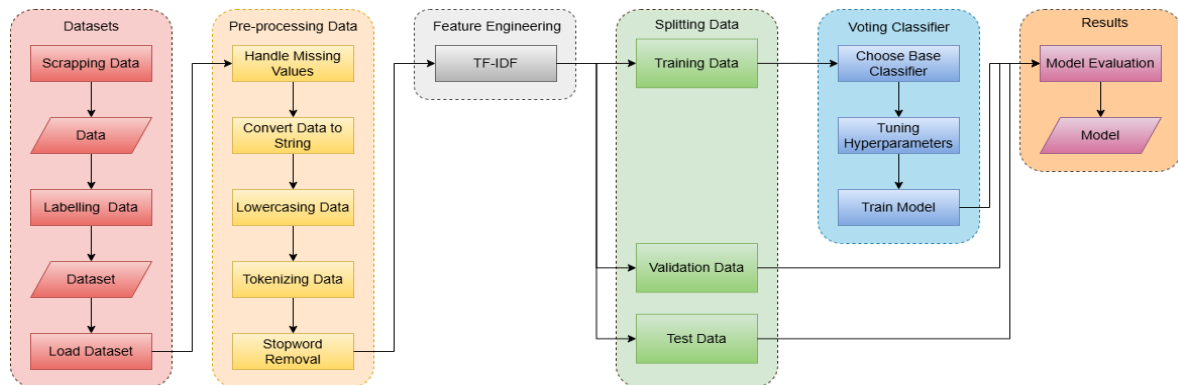


Figure 1. Block diagram illustrating the process of building a model using voting classifier, from data collection and labeling through preprocessing, feature engineering, dataset splitting, model training, and evaluation

#### 3.1. Dataset

The dataset used in this research consists of reviews of the Shopee application from the Google Play Store platform. We collected 1,000 rows of data with columns for username, score, timestamp, and content. Since this raw data required labeling, we added an "identify" column as our target variable, designating each review as either a pain point or non-pain point. The labeling process was conducted with validation from a UX Researcher to ensure accuracy. The final dataset contains a balanced composition of 500 pain point reviews and 500 non-pain point reviews.

As shown in Table 1, the dataset contains examples of both pain points and non-pain points. Pain points typically describe specific issues or difficulties users face, while non-pain points might include general comments or preferences. The dataset undergoes preprocessing stages such as handling missing values,

tokenizing, and stopword removal. After feature engineering, we split the dataset into training data (80%) and test data (20%), with the training data further divided into a 70:30 ratio for actual training and validation [19].

Table 1. Sample dataset of shopee reviews from Google Play Store

Identify	Username	Score	At	Content
Non-Pain Point	Agus Wijaya	2	17/02/2024 06:52	After last 2 latest updates
Non-Pain Point	Black Coffee	2	08/02/2024 14:18	Probably it's great for some people. But i found it very toxic. I spent too much time and money in this app.
..	...	...	...	...
Pain Point	Maura Ega Pramesthi	2	28/03/2024 07:01	It's hard to log in, even though i already put the correct password. Please fix.
Pain Point	Julius Chandra	2	27/03/2024 13:30	How to remove live and video button? It is annoying the app can jump automatically to live/video. Seems there is a bug, once the live/video keep playing even when I have closed the app.

### 3.2. Term frequency-inverse document frequency

As noted by Aizawa [20], TF-IDF is a well-established term weighting scheme commonly employed in information retrieval systems. Bafna *et al.* [21] demonstrated that TF-IDF effectively removes noisy and less relevant data by selecting only the most relevant terms. In our research, we utilize TF-IDF to identify the most relevant words that distinguish between pain point and non-pain point reviews.

TF-IDF consists of two components: TF-IDF. Term frequency represents the occurrence of words in a document, while IDF reflects the number of words in the document [22]. The formulas for TF-IDF are as follows,

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad (1)$$

where  $n_{i,j}$  is the number of occurrences of word  $t_i$  in document  $d_j$ , and  $\sum_k n_{k,j}$  is the total number of occurrences of all words in document  $d_j$ .

$$idf_i = \log \frac{|D|}{|\{j:t_i \in d_j\}|} \quad (2)$$

where  $|D|$  represents the total number of documents, and  $|\{j:t_i \in d_j\}|$  is the occurrence of each word  $t_i$  across all documents. The complete TF-IDF formula combines these components,

$$tfidf_{i,j} = tf_{i,j} \times idf_i \quad (3)$$

### 3.3. Voting classifier

A voting classifier is a meta-classifier that combines multiple ML classifiers to improve classification performance. It represents a type of ensemble learning method that leverages the strengths of different algorithms. There are two primary types of voting classifiers: hard voting and soft voting [7].

Hard voting (majority voting) is a straightforward ensemble method for classification that combines predictions from multiple individual classifiers. The final prediction is determined by the class label that receives the most votes from the individual classifiers. The formula for hard voting is,

$$Y = \text{mode} \{C1(x), C2(x), \dots, Cm(x)\} \quad (4)$$

where  $Y$  represents the result of the hard voting classifier,  $Cm$  is the number of algorithms being compared, and  $x$  is the value from each algorithm being compared.

Soft voting (weighted voting) is a more sophisticated ensemble method that considers the confidence of predictions from individual classifiers. Unlike hard voting, which only counts votes for each class label, soft voting takes into account the estimated probability  $p$  for each classifier. This approach is particularly beneficial when classifiers are well-tuned. The formula for soft voting is,

$$Y = \text{argmax}_i \sum_{j=1}^m W_j P_{ij}, i \in \{0,1\}, [j = 1, 2, \dots, m] \quad (5)$$

where  $Y$  is the result of soft voting classifier then  $i$  for index of class and  $j$  for index of classifier,  $W_j$  is weight that can be given to classifier  $j$ , and  $P_{ij}$  is estimated probability  $p$  for classifier.

To develop the optimal voting classifier model, we compared five different classifiers to determine which combination of three would yield the highest accuracy and F1-score. The classifiers evaluated were: multi-layer perceptron (MLP), XGradient boosting, RF, multinomial NB, and LR. Each of these algorithms brings unique strengths to the ensemble.

### 3.3.1. Multi-layer perceptron classifier

MLPs are neural networks known for their effectiveness in supervised learning tasks using the backpropagation algorithm. These networks typically employ a multi-layer architecture, including an input layer, one or more hidden layers, and an output layer, with each neuron connecting to all neurons in the subsequent layer. MLPs are particularly valuable in our ensemble for their ability to model complex non-linear relationships in text data, and have been successfully implemented in mail classification tasks, even outperforming other algorithms [23].

### 3.3.2. XGradient boosting classifier

XGBoost is a machine learning algorithm that utilizes gradient boosting on decision trees. It is highly sophisticated and powerful, capable of handling irregularities in high-dimensional data. XGBoost offers several advantages such as fast processing, support for diverse input formats, built-in cross-validation, tree pruning, and flexibility in parameter tuning. Moreover, its design incorporates mechanisms to control overfitting, making it more robust and efficient compared to other boosting models [24].

### 3.3.3. Random forest classifier

A RF classifier consists of a collection of features structured classifiers with random labeled feature vectors of a movie by casting a vote for the most popular movie class according to the features input. This ensemble approach leverages the diversity of multiple decision trees to improve classification accuracy and reduce overfitting. In the context of this study, RF demonstrated strong performance, particularly when applied to user reputation, social, and temporal features [17].

### 3.3.4. Multinomial naïve Bayes classifier

The MNB classifier, a variant of NB, is a widely recognized data mining algorithm for classification tasks. Its efficiency primarily stems from the assumption of attribute independence. Despite this strict independence assumption, NB has proven to be a competent classifier in many real-world applications. In our ensemble, MNB contributes its inherent efficiency in handling diverse datasets, leveraging the fundamental efficiency of the NB framework. Its performance can be further enhanced through techniques like attribute selection, which aims to mitigate the effect of the attribute independence assumption [25].

### 3.3.5. Logistic regression classifier

LR is a common and efficient statistical method widely applied to classification problems, including email classification and spam filtering. It excels at predicting binary outcomes based on a set of independent variables by utilizing a logistic activation function to produce probabilistic outputs. LR's simplicity and speed make it a popular choice, particularly for real-time applications. In our ensemble, LR adds value through these probabilistic outputs and its inherent interpretability, which complement the other algorithms' strengths [26]. By combining these diverse algorithms in our voting classifier, we leverage their complementary strengths to create a more robust and accurate pain point identification model. Each algorithm contributes different perspectives on the classification problem, resulting in more balanced and reliable predictions.

## 4. RESULTS AND DISCUSSION

In this section, we evaluate our model using a confusion matrix to calculate accuracy, precision, recall, and F1-score. We also analyze computation duration and compare evaluation results based on varying dataset sizes. Before implementing the voting classifier algorithms, we first determined the optimal dataset size for maximizing model performance. We trained datasets using combinations of three classifier algorithms selected from the five algorithms based on cross-validation values from previous research. The accuracy and F1-score results are presented.

Figure 2 illustrates the relationship between dataset size and model performance. As shown in Figure 2(a), average accuracy generally increases with dataset size, reaching approximately 0.89 for 1,000 data points. Similarly, Figure 2(b) demonstrates that average F1-score follows a similar trend, also reaching about 0.89 with 1,000 data points.

The graphs show nearly identical trends between accuracy and F1-score across different dataset sizes. While there is generally an improvement with increasing dataset size, we observed a slight anomaly when the dataset contained 500 samples; both accuracy and F1-score decreased from 0.868 (at 300 samples)

to 0.864. This anomaly likely resulted from imbalances in pain point and non-pain point distribution within the test results for the 500-sample dataset.

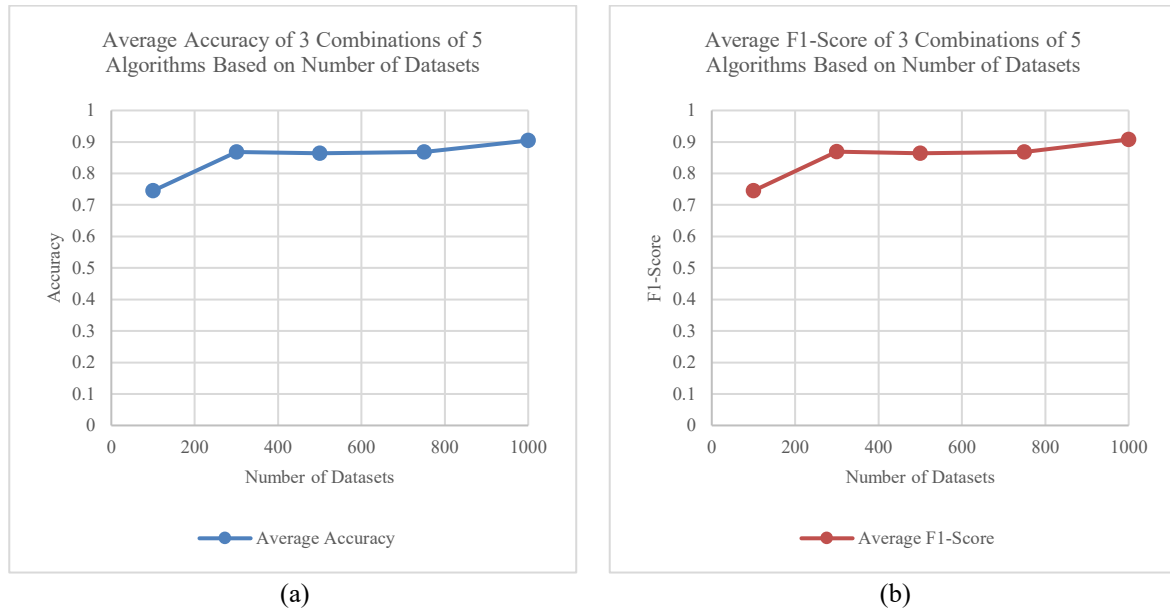


Figure 2. Performance metrics based on dataset size, (a) average accuracy of three classifier combinations with increasing dataset size and (b) average F1-score of three classifier combinations with increasing dataset size

The performance limitations with smaller datasets can be attributed to insufficient training data, causing the model to produce more false positives and false negatives in the confusion matrix. However, the overall results indicate a balanced distribution between positive and negative classes in our dataset. While the improvement is not dramatic, the trend clearly shows that larger datasets tend to yield better average accuracy and F1-scores. Although we also calculated recall and precision, these metrics are not displayed in the graphs since they are already incorporated into the F1-Score using the following formula,

$$F1 - Score = \frac{2 \times recall \times precision}{recall + precision} \quad (6)$$

In addition to performance metrics, we analyzed computation time based on dataset size, as shown in Figure 3. Figure 3 reveals a clear positive correlation between dataset size and computation time. The processing time increases from approximately 15 seconds for 100 samples to 120 seconds for 1,000 samples. This upward trend indicates that larger datasets require significantly more processing time across all stages of model development—from loading datasets and preprocessing to feature engineering, training, testing, validation, and evaluation.

Our model testing followed several schemes. First, based on confusion matrix results during training, we determined that the optimal dataset size was 1,000 samples. We then compared the accuracy and F1-score of models using different combinations of three classifier algorithms selected from the five algorithms tested. All tests used default hyperparameter values to ensure a fair comparison.

Figure 4 compares the accuracy and F1-score results across different voting classifier combinations. All combinations achieved accuracy above 0.85 and F1-scores above 0.80, surpassing the results of previous studies that used neural network algorithms. This demonstrates that the ensemble learning technique can produce better models for this task. The performance differences between combinations (ranging from 0.89 to 0.91) occur because some classifier algorithms are more suitable for our dataset than others.

The analysis identified two top-performing algorithm combinations: i) XGradient boosting, MNB, and LR; and ii) MLP, XGradient boosting, and LR—both achieving an accuracy of 0.915 and an F1-Score of 0.91. However, validation results revealed differences between these combinations, Figure 5 demonstrates that the combination of XGradient boosting, MNB, and LR achieved higher validation accuracy (0.8208) and F1-score (0.82) compared to the combination of MLP, XGradient boosting, and LR (accuracy: 0.7958, F1-

score: 0.80). This difference likely stems from imbalances in the validation data and the MLPs reduced compatibility with our particular dataset. Since both voting classifiers include XGradient boosting and LR, the difference in performance can be attributed to the third algorithm in each combination.

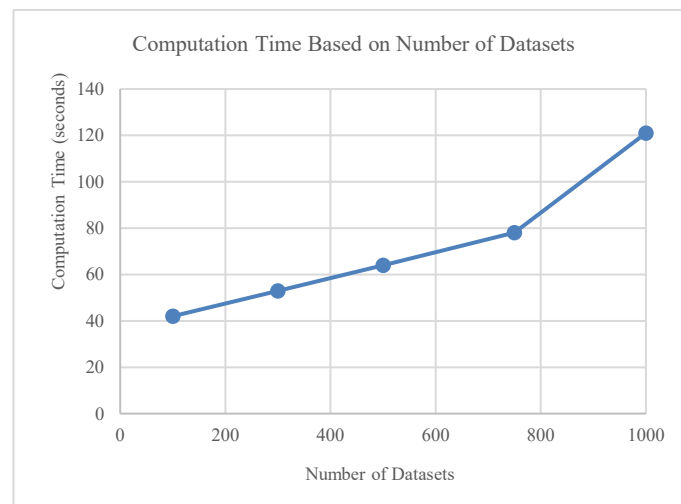


Figure 3. Relationship between computation time (seconds) and dataset size (number of samples)

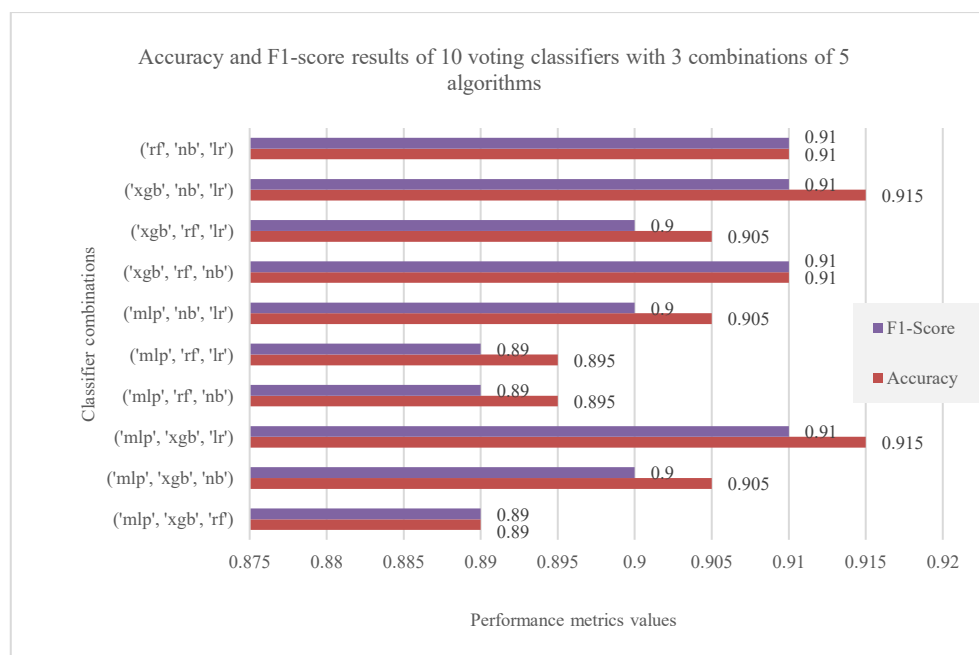


Figure 4. Comparison of accuracy (blue) and F1-score (orange) across 10 different voting classifier combinations

We conclude that the optimal classifier combination is XGradient boosting, MNB, and LR, with a validation accuracy of 0.8208 and F1-score of 0.82. This combination leverages XGradient boosting's ability to handle complex patterns, MNB's strength with text classification, and LR's probabilistic approach. We also compared soft voting and hard voting techniques, Figure 6 reveals identical performance between soft voting and hard voting classifiers in terms of both accuracy and F1-score. This suggests that despite their different mechanisms-soft voting uses weighted probability while hard voting uses direct value comparison-both approaches yield similar results for our dataset. Finally, we compared our voting classifier model with a neural network model trained on the same dataset. As shown in Figure 7, the voting classifier significantly

outperforms the neural network model, achieving 0.9 for both accuracy and F1-score compared to the neural network's approximately 0.82 for both metrics. This 10% performance improvement demonstrates the effectiveness of our ensemble approach for pain point identification in app reviews.

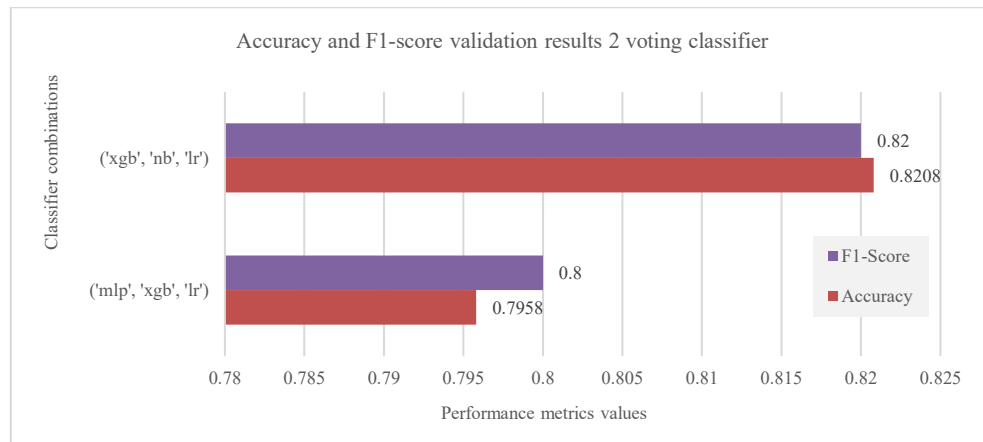


Figure 5. Validation performance comparison between two top voting classifier combinations

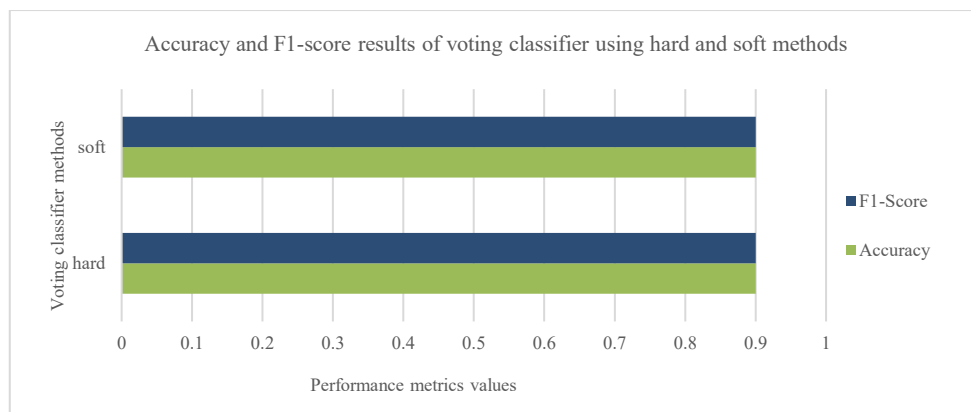


Figure 6. Performance comparison between soft voting and hard voting classifiers

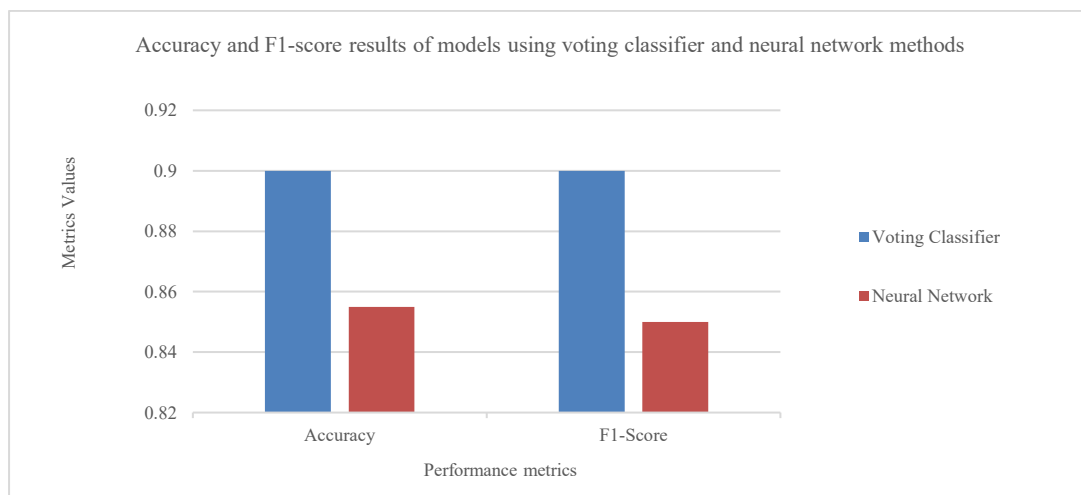


Figure 7. Performance comparison between voting classifier and neural network algorithms



#### 4.1. Model limitations

Despite the strong performance, our model has several limitations:

- Short and ambiguous reviews: the model sometimes struggles with very brief reviews or those with ambiguous language that might contain implied rather than explicit pain points.
- Domain specificity: our model was trained specifically on Shopee app reviews. Its performance might vary when applied to reviews from different app categories (e.g., games, productivity tools) that might have domain-specific vocabulary.
- Language constraints: the current model works primarily with reviews in one language. Multilingual reviews would require additional preprocessing and potentially different modeling approaches.
- Evolving user language: as user language and app terminology evolve over time, the model may require periodic retraining to maintain its accuracy.

#### 5. CONCLUSION

This study demonstrates that the voting classifier approach significantly improves pain point identification accuracy in app reviews, increasing performance from the previous benchmark of 80% to 90%. Our research shows that carefully selected classifier combinations can produce better results, with the XGradient boosting, MNB, and LR combination achieving the highest performance (90% accuracy and F1-score). This superior performance stems from the complementary strengths of these algorithms: XGradient boosting excels at handling complex patterns, MNB is particularly effective for text classification, and LR provides reliable probabilistic outputs. The balance of the dataset is critical for model performance, as demonstrated by our analysis of different dataset sizes. We found that larger, balanced datasets generally yield better results, though computation time increases proportionally with dataset size. Our comparative analysis also revealed that both soft and hard voting techniques produce similar results for our use case, suggesting that either approach can be effectively implemented depending on specific requirements. The voting classifier model significantly outperformed a neural network model trained on the same dataset, achieving approximately 10% higher accuracy and F1-score. This finding highlights the effectiveness of ensemble learning techniques for text classification tasks, particularly for identifying pain points in app reviews.

#### 6. FUTURE WORK

For future research, we recommend: i) expanded dataset diversity: testing the model with reviews from multiple applications across different categories to improve generalizability; ii) multilingual support: extending the model to handle reviews in multiple languages, which would increase its utility in global app markets; iii) fine-tuning hyperparameters: conducting systematic hyperparameter optimization to potentially further improve model performance; iv) real-time implementation: developing a system for real-time pain point detection that could provide immediate feedback to developers; v) deep learning integration: exploring hybrid models that combine the voting classifier approach with deep learning techniques, potentially leveraging transformer models like BERT for improved text understanding; vi) severity classification: extending the model to not only identify pain points but also classify their severity or impact on user experience; and vii) image and video review analysis: investigating methods to analyze pain points expressed in image or video reviews, which are becoming increasingly common on app stores. These directions would build upon our findings and potentially lead to even more accurate and versatile pain point identification systems, ultimately helping developers create better user experiences.

#### ACKNOWLEDGEMENTS

Alhamdulillah. My gratitude also goes to the researchers who have helped me gain insight and new knowledge about the objects and methods I researched.

#### FUNDING INFORMATION

Thank you to Institut Teknologi Nasional Bandung for funding this publication.

#### AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Yusup Miftahuddin	✓	✓		✓	✓	✓		✓		✓	✓	✓	✓	✓
Muhammad Alif Firdaus		✓	✓				✓		✓		✓			

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review &amp; Editing

Vi : Visualization

Su : Supervision

P : Project administration

Fu : Funding acquisition

## CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

## INFORMED CONSENT

We have obtained informed consent from all individuals included in this study.

## DATA AVAILABILITY

For now, the data and application that support the findings of this study are not publicly available due to confidentiality and institutional restrictions. However, further information may be obtained from the corresponding author, [YM], upon reasonable request.





## REFERENCES

- [1] J. Salminen, M. Mustak, J. Corporan, S. G. Jung, and B. J. Jansen, "Detecting pain points from user-generated social media posts using machine learning," *Journal of Interactive Marketing*, vol. 57, no. 3, pp. 517–539, 2022, doi: 10.1177/10949968221095556.
- [2] S. Venkatakrishnan, A. Kaushik, and J. K. Verma, "Sentiment analysis on Google Play Store data using deep learning," pp. 15–30, 2020, doi: 10.1007/978-981-15-3357-0\_2.
- [3] A. Karim, A. Azhari, M. Alruily, H. Aldabbas, S. Brahim Belhaouri, and A. Adil Qureshi, "Classification of Google Play Store application reviews using machine learning," vol. 17, p. 302, Jul. 2020, doi: 10.20944/preprints202007.0646.v1.
- [4] E. J. Lee and S. Y. Shin, "When do consumers buy online product reviews? effects of review quality, product type, and reviewer's photo," *Computers in Human Behavior*, vol. 31, no. 1, pp. 356–366, 2014, doi: 10.1016/j.chb.2013.10.050.
- [5] M. Hadwan, M. Al-Sarem, F. Saeed, and M. A. Al-Hagery, "An improved sentiment classification approach for measuring user satisfaction toward governmental services' mobile apps using machine learning methods with feature engineering and SMOTE technique," *Applied Sciences (Switzerland)*, vol. 12, no. 11, 2022, doi: 10.3390/app12115547.
- [6] K. M. Alomari, H. M. Elsherif, and K. Shaalan, "Arabic tweets sentimental analysis using machine learning," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10350 LNCS, pp. 602–610, 2017, doi: 10.1007/978-3-319-60042-0\_66.
- [7] A. Mahabub, "A robust technique of fake news detection using ensemble voting classifier and comparison with other classifiers," *SN Applied Sciences*, vol. 2, no. 4, 2020, doi: 10.1007/s42452-020-2326-y.
- [8] A. A. Salamai, E. S. M. El-Kenawy, and I. Abdelhameed, "Dynamic voting classifier for risk identification in supply chain 4.0," *Computers, Materials and Continua*, vol. 69, no. 3, pp. 3749–3766, 2021, doi: 10.32604/cmc.2021.018179.
- [9] R. M. Amir Latif, M. Talha Abdullah, S. U. Aslam Shah, M. Farhan, F. Ijaz, and A. Karim, "Data scraping from google play store and visualization of its content for analytics," *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies, iCoMET 2019*, 2019, doi: 10.1109/ICOMET.2019.8673523.
- [10] N. Bensouda, S. El Fkihi, and R. Faizi, "A novel ensemble model for detecting fake news," *IAES International Journal of Artificial Intelligence*, vol. 13, no. 1, pp. 1160–1171, 2024, doi: 10.11591/ijai.v13.i1.pp1160-1171.
- [11] E. Elsaed, O. Ouda, M. M. Elmogy, A. Atwan, and E. El-Daydamony, "Detecting fake news in social media using voting classifier," *IEEE Access*, vol. 9, pp. 161909–161925, 2021, doi: 10.1109/ACCESS.2021.3132022.
- [12] T. B. Chandra, K. Verma, B. K. Singh, D. Jain, and S. S. Netam, "Coronavirus disease (COVID-19) detection in chest X-ray images using majority voting based classifier ensemble," *Expert Systems with Applications*, vol. 165, 2021, doi: 10.1016/j.eswa.2020.113909.
- [13] N. Rai, N. Kaushik, D. Kumar, C. Raj, and A. Ali, "Mortality prediction of COVID-19 patients using soft voting classifier," *International Journal of Cognitive Computing in Engineering*, vol. 3, pp. 172–179, 2022, doi: 10.1016/j.ijcce.2022.09.001.
- [14] V. Alimisis, V. Mouzakis, G. Gennis, E. Tsouvalas, C. Dimas, and P. P. Sotiriadis, "A hand gesture recognition circuit utilizing an analog voting classifier," *Electronics (Switzerland)*, vol. 11, no. 23, 2022, doi: 10.3390/electronics11233915.
- [15] H. B. Kibria, M. Nahiduzzaman, M. O. F. Goni, M. Ahsan, and J. Haider, "An ensemble approach for the prediction of diabetes mellitus using a soft voting classifier with an explainable AI," *Sensors*, vol. 22, no. 19, 2022, doi: 10.3390/s22197268.
- [16] F. Balouchzahi, H. L. Shashirekha, and G. Sidorov, "HSSD: hate speech spreader detection using N-Grams and voting classifier," in *CEUR Workshop Proceedings*, 2021, vol. 2936, pp. 1829–1836.
- [17] M. S. Faisal, A. Rizwan, K. Iqbal, H. Fasihuddin, A. Banjar, and A. Daud, "Prediction of movie quality via adaptive voting classifier," *IEEE Access*, vol. 10, pp. 81581–81596, 2022, doi: 10.1109/ACCESS.2022.3195228.
- [18] D. S. Khafaga *et al.*, "Voting classifier and metaheuristic optimization for network intrusion detection," *Computers, Materials & Continua*, vol. 74, no. 2, pp. 3183–3198, 2023, doi: 10.32604/cmc.2023.033513.





- [19] I. Muraina, "Ideal dataset splitting ratios in machine learning algorithms: general concerns for data scientists and data analysts," in *7th international Mardin Artuklu scientific research conference*, 2022, pp. 496–504.
- [20] A. Aizawa, "An information-theoretic perspective of tf-idf measures," *Information Processing and Management*, vol. 39, no. 2003, pp. 45–65, 2002.
- [21] P. Bafna, D. Pramod, and A. Vaidya, "Document clustering: TF-IDF approach," *International Conference on Electrical, Electronics, and Optimization Techniques, ICEEOT 2016*, pp. 61–66, 2016, doi: 10.1109/ICEEOT.2016.7754750.
- [22] C. Z. Liu, Y. X. Sheng, Z. Q. Wei, and Y. Q. Yang, "Research of text classification based on improved TF-IDF algorithm," in *2018 IEEE International Conference of Intelligent Robotic and Control Engineering, IRCE 2018*, 2018, pp. 69–73, doi: 10.1109/IRCE.2018.8492945.
- [23] A. Occhipinti, L. Rogers, and C. Angione, "A pipeline and comparative study of 12 machine learning models for text classification," *Expert Systems with Applications*, vol. 201, 2022, doi: 10.1016/j.eswa.2022.117193.
- [24] A. Bansal and S. Kaur, "Extreme gradient boosting based tuning for classification in intrusion detection systems," in *Advances in Computing and Data Sciences*, Singapore: Springer, 2018, pp. 372–380, doi: 10.1007/978-981-13-1810-8\_37.
- [25] S. Chen, G. I. Webb, L. Liu, and X. Ma, "A novel selective naïve Bayes algorithm," *Knowledge-Based Systems*, vol. 192, p. 105361, 2020, doi: 10.1016/j.knsys.2019.105361.
- [26] B. K. Dedeturk and B. Akay, "Spam filtering using a logistic regression model trained by an artificial bee colony algorithm," *Applied Soft Computing Journal*, vol. 91, 2020, doi: 10.1016/j.asoc.2020.106229.

## BIOGRAPHIES OF AUTHORS



**Yusup Miftahuddin**     earned his S.Kom degree from Institut Teknologi Nasional, Bandung, in 2006. He also received his M.T. degree from Institut Teknologi Bandung, Bandung, in 2012. He is currently a lecturer at the Department of Informatics, Institut Teknologi Nasional. He is also a researcher in artificial intelligence and data science at Institut Teknologi Nasional since January 2013. His research includes classification of roasted coffee beans using principal component analysis and random forests, identification of glaucoma using relevant vector machines, application of data standardization and multilayer perceptron on phishing website identification, and detection of cyberbullying comments on English social media using naïve Bayes classification. He has published more than 30 papers in international journals and conferences. He can be contacted at email: yusufm@itenas.ac.id.



**Muhammad Alif Firdaus**     is a fresh graduate of the informatics Department who completed his bachelor's degree from Institut Teknologi Nasional, Bandung, in 2024. He has participated in international certification from Huawei in the field of artificial intelligence (HCAI), and also received several machine learning certificates from Dicoding Indonesia. Currently, he works as a Fullstack Engineer in an IT Consultant Office. He can be contacted at email: alifmuhammadfirdaus1@gmail.com.