

Sentiment Analysis of Flip Application Reviews Using DBSCAN Outlier Removal and SVM

Hidayatul Ichwan^{1)*}, Fajar Mahardika²⁾, Ratih³⁾, Riki Aldi Pari⁴⁾

¹⁾Department of Informatics Engineering, Sekolah Tinggi Manajemen Informatika dan Komputer Jakarta (STMIK Jakarta), Jakarta, Indonesia

²⁾Department of Computer and Business, Informatics Engineering Program, Politeknik Negeri Cilacap, Cilacap, Indonesia

³⁾Department of Computer and Business, Cyber Security Engineering Program, Politeknik Negeri Cilacap, Cilacap, Indonesia

⁴⁾Cybersecurity Engineering Study Program, Piksi Input Polytechnic, Serang, Banten, Indonesia

¹⁾hidayatul_ichwan@stmik.jayakarta.ac.id, ²⁾fajarmahardika@pnc.ac.id, ³⁾rikialdipari@gmail.com

Submitted : Aug 7, 2026 | **Accepted :** Sep 17, 2026 | **Published :** Oct 1, 2026

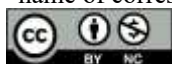
Abstract: Flip is an Indonesian financial technology application offering free interbank transfers, whose Google Play Store review column has become a large unstructured record of user experience that a development team cannot read manually. Such reviews are noisy, containing slang, repeated characters, and very short comments that behave as outliers. This study measures whether removing DBSCAN-detected outliers from the training data improves a support vector machine that assigns Flip reviews to three rating-derived categories. From 20,000 collected reviews, 15,241 remained after duplicate removal and preprocessing. The corpus was split once into stratified training and test partitions before any fitting, so that TF-IDF, the truncated singular value decomposition and DBSCAN were fitted on the training partition alone and both scenarios were evaluated on one identical held-out test set of 3,049 documents, repeated over five random seeds. Because the k-distance curve shows no pronounced knee, eps was fixed at the ninetieth percentile of that distribution, and eps and minPts were varied over a grid of nine combinations. Accuracy was 0.8023 without filtering and 0.8033 with it, a paired difference of +0.0010 with a 95 percent confidence interval of [-0.0017, +0.0038] and $p = 0.347$; macro F1-score moved from 0.6390 to 0.6402. No combination in the grid produced a distinguishable improvement. Of forty flagged documents inspected manually, none was genuine noise, and 91 percent of the flagged documents belonged to the majority positive class. The results give no evidence that density-based filtering benefits short-text classification.

Keywords: class imbalance; DBSCAN; Flip application; outlier detection; sentiment analysis; support vector machine; TF-IDF

INTRODUCTION

The adoption of financial technology services in Indonesia has grown rapidly, and much of the resulting user experience is recorded in the review column of the Google Play Store. Flip is a representative case: an application whose principal appeal is free interbank transfer, and whose review column accumulates a continuous stream of opinion about transaction speed, fees, refunds, and customer service. Flip passed ten million users in the first half of 2022 (Antara News, 2022) and operates under a licence issued by Bank Indonesia (PT Fliptech Lentera Inspirasi Pertiwi, 2026). The volume of feedback

*name of corresponding author



This is anCreative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

has long passed the point at which manual reading is feasible, and averaging the star rating hides the reason behind the score. Because the service moves customer funds, a complaint about a delayed transfer is a consumer-protection issue rather than a usability one, so accurate automatic reporting of negative opinion is an operational requirement.

Among the classification algorithms applied to this problem, the support vector machine remains the reference choice for Indonesian application reviews represented by TF-IDF features (Cortes & Vapnik, 1995; Joachims, 1998). Its performance depends on the quality of the training data, and application-store text is irregular: slang, repeated characters and one-word comments are common, and in a sparse high-dimensional space such documents occupy low-density regions that can distort the decision boundary. A common response is to filter before training. Density-based spatial clustering of applications with noise (Ester et al., 1996) labels points in low-density regions as noise without requiring the number of clusters in advance. Whether such filtering actually helps a short-text classifier is an empirical question the Indonesian literature has not tested directly.

The novelty of this study lies in three points. First, DBSCAN is employed as a preprocessing filter for a supervised classifier rather than as a clustering method in its own right, which is the role it plays in almost all Indonesian sentiment analysis literature. Second, ϵ is not assigned by trial and error but fixed by an explicit rule applied to the distribution of k -nearest-neighbour cosine distances over TF-IDF vectors. As reported in the results, that curve rises gradually and shows no pronounced knee, so a curvature-based threshold cannot be located; ϵ is therefore set at the ninetieth percentile of the distribution, which is reproducible and stated in advance but fixes the size of the outlier budget by construction. Because that is a convention, the sensitivity of the outcome to both ϵ and minPts is examined over a grid. Third, the effect of filtering is measured against an identically configured baseline under a paired protocol: the test partition is fixed before any fitting, the vectoriser and the filter are fitted on the training partition alone, and both models are evaluated on the same unfiltered held-out documents, with macro-averaged F1-score reported so that the minority neutral class is not concealed by the majority class.

The objective is therefore to measure, on a corpus of Flip reviews from the Google Play Store, whether removing DBSCAN-detected outliers from the training data before fitting a support vector machine changes performance on a fixed held-out test set, and in which direction. Two qualifications are stated at the outset. The labels are derived from star ratings and are therefore a proxy for the sentiment expressed in the text; the gap between the two is measured on a manually annotated sample, and to the extent that it is large the task is better described as predicting the rating category from the review text. The comparison is estimated from repeated stratified splits, so the reported difference can be placed against its own sampling variability rather than read as an exact effect.

LITERATURE REVIEW

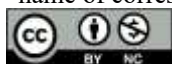
Sentiment Analysis of Indonesian Application Reviews

Sentiment analysis is the computational study of opinion and subjectivity in text, normally treated at document level as supervised classification (Pang & Lee, 2008; Liu, 2012; Medhat et al., 2014). TF-IDF weighting assigns a higher weight to terms frequent in a document but rare across the corpus (Salton & Buckley, 1988), producing the sparse vectors on which the support vector machine operates; the classifier seeks the hyperplane of maximum margin (Cortes & Vapnik, 1995) and remains effective in high dimensions (Joachims, 1998). Many Indonesian studies apply this configuration to Google Play Store reviews; Madjid et al. (2023), Mustakim and Priyanta (2022), Setyani et al. (2026) and Hamid et al. (2025) are compared in Table 1, while Purwanto et al. (2026) used recurrent networks and Purwanto, Mahardika and Nugroho (2026) applied IndoBERT with weighted loss. All limit preprocessing to linguistic normalisation.

Density-Based Outlier Detection

DBSCAN was introduced by Ester et al. (1996) as a clustering algorithm that groups points in dense regions and labels the remainder as noise, governed by a radius ϵ and a minimum number of neighbours minPts . Schubert et al. (2017) emphasised that its output depends critically on these parameters and that ϵ should be derived from the distance distribution rather than guessed.

*name of corresponding author



This is anCreative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Alternatives include the local outlier factor (Breunig et al., 2000) and the isolation forest (Liu et al., 2008), compared by Ratih et al. (2026); the taxonomy is surveyed by Chandola et al. (2009) and Aggarwal (2017). From a different direction, the edited nearest neighbour rule (Wilson, 1972) removes instances whose labels disagree with their neighbours, and Frenay and Verleysen (2014) document how label noise, as distinct from feature-space anomaly, degrades performance. A document unusual in feature space is not necessarily one whose label is wrong.

State of the Art

Table 1 summarises the most relevant recent studies on sentiment classification of Indonesian application reviews, comparing the data examined, the methods applied, the treatment of anomalous observations, and the reported outcome together with its limitation.

Table 1. State of the art of sentiment analysis on Indonesian application reviews

Author (Year)	Object and data	Method	Outlier handling	Outcome and limitation
Mustakim & Priyanta (2022)	KAI Access reviews, aspect-based	TF-IDF with naïve Bayes and SVM	None	SVM competitive per aspect; anomalous reviews retained
Madjid et al. (2023)	Mobile banking application reviews	TF-IDF with SVM and naïve Bayes	None	SVM outperforms naïve Bayes; minority class not reported
Hamid et al. (2025)	Indonesian mobile government applications	Sentiment analysis with topic modelling	None	Polarity linked to service topics; no noise filtering stage
Setyani et al. (2026)	Jamsostek Mobile reviews	TF-IDF with SVM and naïve Bayes	None	Accuracy reported on imbalanced data; neutral class not isolated
This study	15,318 Flip reviews, three classes	TF-IDF unigram and bigram (4,000 features) with linear SVM, balanced class weights, fitted on training data only	DBSCAN on training data only, in a 100-component SVD space with Euclidean distance; eps at the 90th percentile of the k-distance distribution, with a sensitivity grid	No effect detectable against a matched baseline on one identical held-out test set over five seeds; macro F1 reported

Two patterns are visible in Table 1. The first is methodological convergence: TF-IDF with a support vector machine remains the reference configuration, with accuracy generally between seventy and ninety percent depending on whether the neutral class is retained. The second is a shared omission: none of these studies applies noise filtering to the training data, and none reports macro-averaged performance, so it cannot be determined whether the models handle ambivalent reviews or merely reproduce the majority class. What is missing is a direct measurement of the interaction between the two, using a reproducible parameter selection procedure and an evaluation that reports the minority class explicitly. This study supplies that measurement.

METHOD

This research follows a quantitative experimental design of eight sequential stages. The experiment is a paired comparison between two scenarios differing only in the presence of the DBSCAN filtering

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

stage. The corpus is divided into training and test partitions before any model component is fitted; the TF-IDF vectoriser, the truncated singular value decomposition and the DBSCAN filter are fitted on the training partition alone; filtering removes documents from the training partition only; and both classifiers are evaluated on the same unfiltered held-out test set. No test document is removed in either scenario, so any difference is attributable to the training data rather than to a change in the evaluation sample. The procedure is repeated over five stratified splits with different random seeds, and each metric is reported as a mean with its standard deviation. Implementation used Python and scikit-learn (Pedregosa et al., 2011).

Data Collection

The dataset consists of Indonesian-language user reviews of the Flip application retrieved from the Google Play Store through automated scraping of the public review list. Each record contains the review text, the star rating on a scale from one to five, and the timestamp. After duplicate removal, 15,318 reviews were retained; only the text and the rating are used in modelling.

To support replication, the following configuration is reported. Reviews were retrieved with the google-play-scraper library for the application identifier id.flip, language and country set to Indonesian, sorted by newest, until twenty thousand records were collected; the retained reviews were posted between 27 February 2024 and 5 August 2026. Records with an identical review text were treated as duplicates and removed, leaving 15,318 reviews. Slang normalisation used an eighty-four entry dictionary released with the code, and stopword removal used the Sastrawi list extended with nya, yg, the, and, aplikasi and flip. One consequence must be stated plainly: the slang dictionary maps gak, ga, nggak, tdk and tak onto tidak, and tidak and belum are themselves Sastrawi stopwords, so the reference pipeline removes negation, reducing lama dan tdk gratis to lama gratis. A variant retaining tidak, bukan, belum, jangan, kurang and tanpa is reported in Table 9. The TF-IDF vectoriser was fitted on the training partition only with ngram_range = (1, 2), min_df = 5, max_df = 0.9 and max_features = 4000. All stochastic components used the seeds 42 to 46. The dataset, dictionary, notebook and analysis code are at [repository URL].

Text Preprocessing

Review text from an application store is highly irregular, so preprocessing runs in five stages: cleaning removes URLs, mentions, hashtags, emoji, punctuation and digits and collapses runs of three or more identical characters; case folding converts the text to lower case; slang normalisation maps informal forms to their standard equivalents using the dictionary cited above; stopword removal discards high-frequency function words while retaining the negation particles, since deleting them would invert polarity; and stemming reduces each remaining token to its root using the Sastrawi stemmer. Documents left empty by this pipeline are discarded.

Sentiment Labelling

Sentiment labels are derived from the star rating rather than from full manual annotation, which is the conventional approach in this line of research because it produces a consistent label at scale without annotator disagreement. Reviews rated one or two stars are labelled negative, reviews rated three stars are labelled neutral, and reviews rated four or five stars are labelled positive. The resulting distribution is strongly imbalanced and is described in the results section.

A star rating is, however, a proxy for the sentiment expressed in the text rather than an annotation of it, and the mismatch is expected to be largest in the three-star band, where a review often combines praise with a complaint. To measure this gap rather than assume it away, a random sample of 400 preprocessed reviews, stratified over the five rating values, is annotated independently by two annotators who see the text without the star rating and apply a written guideline with the three categories used here. Agreement is reported as Cohen's kappa, disagreements are resolved by discussion to form a reference label, and that label is compared with the rating-derived label, overall and within the three-star band. The rating-derived labels are retained for modelling so that the whole corpus can be used, but the measured agreement bounds what the classifier can be said to recover.

Feature Extraction

Each preprocessed document is converted into a numerical vector using term frequency and inverse document frequency weighting (Manning et al., 2008), computed as shown in (1), where $tf(t,d)$ is the frequency of term t in document d , N is the total number of documents, and $df(t)$ is the number of documents containing t .

$$w(t,d) = tf(t,d) \times \log(N / df(t)) \quad (1)$$

Both unigrams and bigrams are extracted, so that expressions whose meaning depends on word order are preserved as single features. The resulting document vectors are sparse and high-dimensional, and each vector is normalised to unit length so that document length does not dominate the similarity computation.

Outlier Detection Using DBSCAN

DBSCAN separates the corpus into core points, border points, and noise points. A point is a core point when at least $minPts$ documents lie within a radius of eps around it, a border point lies within eps of a core point without satisfying the density condition itself, and any point that is neither is labelled as noise. Because the vectors are length-normalised and sparse, cosine distance is used as the metric, defined in (2).

$$d(x, y) = 1 - (x \cdot y) / (||x|| ||y||) \quad (2)$$

DBSCAN is not applied to the TF-IDF matrix directly. A 4,000-dimensional sparse space makes density estimates unstable, so the training matrix is first reduced with truncated singular value decomposition to one hundred components, which retain 33.3 percent of the variance, and DBSCAN is run on that representation with Euclidean distance. Both the decomposition and the filter are fitted on the training partition alone. $minPts$ is set to ten, a practical convention for short-text corpora rather than a value derived from the data, and its influence is tested in the sensitivity analysis below. For every training document the distance to its tenth nearest neighbour is computed and the distances are sorted. As reported in the results, the resulting curve rises gradually and shows no pronounced knee, so no unique curvature-based threshold can be identified, and the ninetieth percentile of the sorted distances is adopted as eps . The rule is explicit and reproducible, but it fixes in advance the proportion of documents lying beyond the threshold, so the outlier budget follows from the percentile rather than from structure discovered in the data. Documents labelled as noise are removed from the training partition in the second scenario; the test partition is never inspected by DBSCAN.

Because both parameters are conventions rather than quantities estimated from the data, a sensitivity analysis is carried out instead of reporting a single setting. eps takes the eightieth, ninetieth and ninety-fifth percentiles of the k -distance distribution computed on the training partition, and $minPts$ takes the values five, ten and twenty. For each of the nine combinations the number of training documents removed and the accuracy and macro F1-score on the fixed test set are recorded, which separates the conclusion from the values chosen by the authors and makes visible whether any setting yields an improvement.

Classification Using Support Vector Machine

The classifier is a support vector machine with a linear kernel and the one-versus-rest strategy, trained with `class_weight` set to balanced. The corpus is divided into training and testing subsets in a ratio of eighty to twenty using stratified sampling, before the vectoriser, the decomposition and the filter are fitted, so that both scenarios are evaluated on the same test documents and no statistic from the test partition informs the feature weights. Within each repetition the scenarios differ only in whether the flagged training documents are dropped. The procedure is repeated for five random seeds.

$$\min \frac{1}{2} ||w||^2 + C \sum \xi_i \quad \text{subject to} \quad y_i(w \cdot x_i + b) \geq 1 - \xi_i, \xi_i \geq 0 \quad (3)$$

The classifier is a support vector machine with a linear kernel and the one-versus-rest strategy, trained with `class_weight` set to balanced. The corpus is divided into training and testing subsets in a ratio of eighty to twenty using stratified sampling, before the vectoriser, the decomposition and the filter are fitted, so that both scenarios are evaluated on the same test documents and no statistic from the test

partition informs the feature weights. Within each repetition the scenarios differ only in whether the flagged training documents are dropped. The procedure is repeated for five random seeds.

Evaluation Metrics

Model performance is measured from the confusion matrix using accuracy, precision, recall, and F1-score, defined in (4) to (7).

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (4)$$

$$Precision = TP / (TP + FP) \quad (5)$$

$$Recall = TP / (TP + FN) \quad (6)$$

$$F1 = 2 \times (Precision \times Recall) / (Precision + Recall) \quad (7)$$

Because the class distribution is severely imbalanced, accuracy alone would reward a model that predicts the majority class, so the macro-averaged F1-score is reported alongside it (Sokolova & Lapalme, 2009). The macro average weights every class equally and therefore exposes the behaviour of the minority neutral class. Per-class precision, recall, and F1-score and the confusion matrix are reported for both scenarios.

$$Macro-F1 = (1/k) \sum F1_i \quad (8)$$

Because the two scenarios are evaluated on the same test documents within each repetition, the predictions are paired. The difference in accuracy and in macro F1-score is summarised by its mean across the five repetitions with a 95 percent confidence interval computed from the per-repetition differences, and a paired t-test on those differences is reported. No claim of improvement or deterioration is made for a difference whose interval contains zero.

RESULT

Dataset Characteristics

The distribution of star ratings in the collected dataset is presented in Table 2. It is strongly polarised: five-star reviews account for more than half of the corpus, one-star reviews form the second largest group, and the intermediate ratings of two, three, and four stars together account for less than seventeen percent. This shape is typical of voluntary application-store feedback, where users who write a review are usually either satisfied or frustrated, and it is the source of the class imbalance that dominates the classification results reported below.

Table 2. Distribution of star ratings and the derived sentiment labels

Rating (stars)	Number of reviews	Percentage
1	4,009	26.17%
2	713	4.65%
3	890	5.81%
4	966	6.31%
5	8,740	57.06%
Total	15,318	100.00%

Mapping the ratings to sentiment classes yields a strongly imbalanced distribution: 9,706 positive, 4,722 negative, and only 890 neutral reviews, a ratio of roughly eleven to one between the largest and smallest class, which becomes the central difficulty in classification. Table 3 traces the class distribution through every stage of the pipeline. Preprocessing emptied 77 documents, 67 of them positive, leaving 15,241 for modelling; the stratified split produced 12,192 training and 3,049 test documents, and the filter removed a further 338 from the training partition alone.

Table 3. Class distribution of the corpus at each processing stage

Stage	Positive	Neutral	Negative	Total
Collected reviews, after removal of duplicates	9,706	890	4,722	15,318
Documents entering the modelling stage	9,639	890	4,712	15,241
Training partition (80%), before filtering	7,711	712	3,769	12,192
Test partition (20%), identical in both scenarios	1,928	178	943	3,049
Flagged as outliers in the training partition	307	6	25	338
Training partition after DBSCAN filtering	7,404	706	3,744	11,854

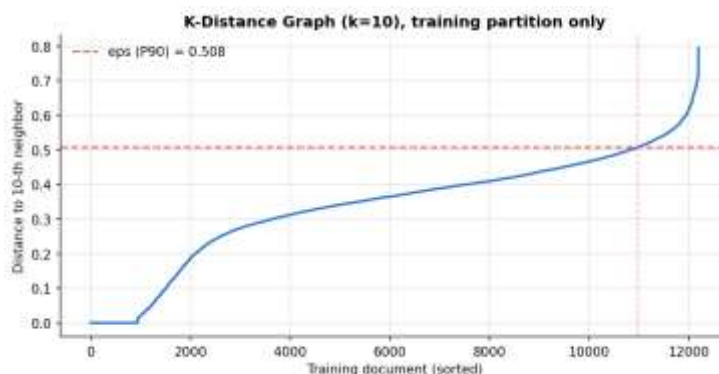
The vocabulary of each class after preprocessing overlaps considerably. The positive class is dominated by terms of assistance and approval such as bantu, bagus, cepat, and mudah, together with biaya admin. The neutral class is dominated by descriptive and temporal terms such as transaksi, lama, and sekarang, indicating that three-star reviews mostly report a state of affairs rather than an evaluation. The negative class shares much of the positive high-frequency vocabulary and is distinguished mainly by kecewa, lambat, refund, and layan langgan. This overlap is the first indication that the three classes are not linearly separable on lexical features alone, and that the neutral class in particular lacks a distinctive lexicon.

Label Validation

The sample for that validation has been drawn and the guideline written, but the annotation had not been completed at the time of submission, so no agreement figure can be given here; Cohen's kappa, the agreement with the rating-derived labels and the agreement within the three-star band will be supplied in the next revision. Until they are, the classifier evaluated below should be read as predicting the rating category from the review text rather than as recovering textual sentiment. One observation from the outlier inspection already indicates the gap is real: among forty flagged documents, one carried a five-star rating on the single word Lamaaaaaaaaa, a complaint about slowness.

Parameter Selection and Outlier Detection

The k-distance graph computed on the training partition is shown in Fig. 1. The curve uses $k = 10$ on the hundred-component representation and rises slowly before turning upward near the right-hand end. The rise is gradual rather than abrupt and the curve shows no pronounced knee, so no curvature-based value of eps can be read off the graph. The ninetieth percentile of the sorted tenth-neighbour distances falls at 0.508 in the first repetition and at 0.510 on average across the five, with a standard deviation of 0.002.

Fig. 1 K-distance graph with $k = 10$, computed on the training partition only

*name of corresponding author



This is anCreative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Applying DBSCAN with the reference parameters to the 12,192 training documents produced the outcome in Table 4: 338 documents, equal to 2.77 percent, were labelled noise and no test document was examined or removed. Table 5 gives the distribution of the flagged documents across classes, and it is the opposite of what an account based on class imbalance would predict. Of the 338, 307 are positive, 25 negative and only 6 neutral, so the filter removed 3.98 percent of the positive class but 0.84 percent of the neutral class. The minority class was the least affected, not the most.

Table 4. DBSCAN configuration and outlier detection result on the training partition

Parameter or output	Value
Representation	Truncated SVD, 100 components (33.3% of variance), Euclidean distance; fitted on training data only
minPts (k), reference setting	10
Rule for eps	Ninetieth percentile of the k-distance distribution (no pronounced knee)
eps, reference setting	0.508 (mean 0.510 ± 0.002 over five seeds)
Outliers removed from training	338 (2.77%); 343 ± 11 over five seeds

Table 5. Distribution of the DBSCAN-flagged documents across sentiment classes in the training partition

Class	Training documents before filtering	Flagged as outlier	Share of the class flagged	Remaining in training
Positive	7,711	307	3.98%	7,404
Neutral	712	6	0.84%	706
Negative	3,769	25	0.66%	3,744
Total	12,192	338	2.77%	11,854

Table 6 reports a manual inspection of a random sample of forty flagged documents. Each was read by the authors and classified as genuine noise, that is spam, repeated characters or a comment carrying no opinion, or as a valid review whose vocabulary is simply rare in this corpus. None of the forty was genuine noise. Thirty-nine were ordinary short reviews, mostly brief praise, and one was a valid review whose text contradicted its star rating. The documents the filter discards are therefore not noise in any useful sense.

Table 6. Manual inspection of a sample of the documents flagged as outliers

No.	Review text	Rating-derived label	Assessment
1	sangat membantu dan mudah digunakan. Proses refund saat ada kesalahan top-up pun cepat. terimakasih flip	Positive	Valid review, rare word combination
2	Terima kasih Aplikasi nya keren dan amanah..	Positive	Valid review, very short
3	irim uang paling cepet gratis biaya lagi mantep emng flip sukses selalu Filip	Positive	Valid review, informal spelling
4	Lamaaaaaaaaaa	Positive	Valid text, but contradicts its five-star rating

Table 7 reports the sensitivity of the filtering stage to its two parameters, for the ninetieth and ninety-fifth percentiles of the k-distance distribution; the eightieth percentile was also tested and removed between 579 and 1,354 training documents, reaching accuracies of 0.7852 to 0.7914 and macro F1-

scores of 0.6315 to 0.6388. Across all nine combinations the number of documents removed varies by a factor of twenty, yet accuracy stays between 0.785 and 0.792 and macro F1-score between 0.632 and 0.641, against a baseline of 0.788 and 0.633. The spread across the whole grid is of the same order as the variation between random splits, so no combination produces an improvement distinguishable from noise.

Table 7. Sensitivity of the outlier count and of classification performance to eps and minPts

eps (percentile of k-distance)	minPts	Training documents removed	Accuracy	Macro F1-score
No filtering (baseline)	—	0	0.7878	0.6333
90th	5	587	0.7911	0.6383
95th	5	247	0.7904	0.6391
90th	10	338	0.7921	0.6409
95th	10	146	0.7898	0.6370
90th	20	169	0.7901	0.6373
95th	20	65	0.7888	0.6351

Classification Performance

The support vector machine was trained under both scenarios within each repetition, with the linear kernel, balanced class weights and the 4,000-feature TF-IDF representation described in the method. Without filtering it saw all 12,192 training documents; with filtering, 11,854. Both were evaluated on the same 3,049 held-out documents, never seen by the vectoriser, the decomposition or the filter. Per-class results appear in Table 8 and the confusion matrices in Fig. 2.

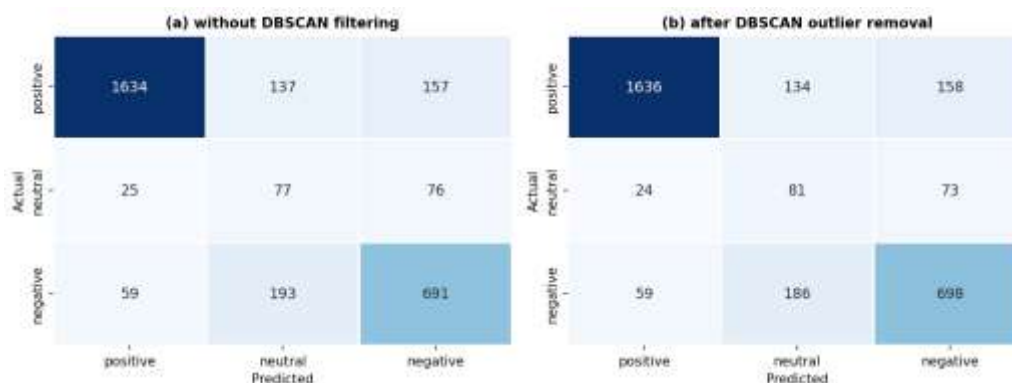


Fig. 2 Confusion matrices of the support vector machine on the identical held-out test set, (a) without and (b) with DBSCAN outlier removal

Table 8 sets the per-class metrics of the two scenarios side by side for the first repetition, and the two are almost identical. The positive class is recovered well in both and the negative class moderately, while the neutral class is barely recovered: precision near 0.19 to 0.20 against recall of 0.43 to 0.46, so its prediction region is large and mostly filled with documents from the other two classes.

Table 8. Per-class performance of both scenarios on the identical held-out test set of 3,049 documents

Class	Scenario	Precision	Recall	F1-score
Positive	Without DBSCAN	0.951	0.848	0.896
	With DBSCAN	0.952	0.849	0.897
Neutral	Without DBSCAN	0.189	0.433	0.263
	With DBSCAN	0.202	0.455	0.280
Negative	Without DBSCAN	0.748	0.733	0.740

*name of corresponding author



This is anCreative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Class	Scenario	Precision	Recall	F1-score
Macro average	With DBSCAN	0.751	0.740	0.746
	Without DBSCAN	0.629	0.671	0.633
	With DBSCAN	0.635	0.681	0.641

The two confusion matrices in Fig. 2 show the same pattern: the neutral column absorbs 137 positive and 193 negative documents in the baseline against 134 and 186 after filtering, and the largest single error in both is negative reviews predicted as neutral.

The comparison is presented in Table 9. Both scenarios are evaluated on the same 3,049 held-out documents in every repetition, so the predictions are paired and the difference can be read as an effect of the training data alone. Averaged over five stratified splits, accuracy was 0.8023 ± 0.0103 without filtering and 0.8033 ± 0.0096 with it, and macro F1-score 0.6390 ± 0.0074 against 0.6402 ± 0.0091 . The mean paired difference was $+0.0010$ in accuracy, with a 95 percent confidence interval of $[-0.0017, +0.0038]$ and $p = 0.347$, and $+0.0011$ in macro F1-score, interval $[-0.0050, +0.0073]$ and $p = 0.636$. Both intervals contain zero, so the data provide no evidence that filtering changes performance in either direction. The lower rows report the variant in which negation particles are retained during preprocessing; that single change raises accuracy by 0.0071 and macro F1-score by 0.0050, which is larger than anything the filtering stage achieved.

Table 9. Comparison of support vector machine performance with and without DBSCAN outlier removal, on an identical held-out test set

Scenario	Training data	Test data	Accuracy (mean $\pm$ sd)	Macro F1-score (mean $\pm$ sd)
Without DBSCAN	12,192	3,049	0.8023 ± 0.0103	0.6390 ± 0.0074
With DBSCAN	11,854	3,049	0.8033 ± 0.0096	0.6402 ± 0.0091
Paired difference (95% CI)	—	—	$+0.0010 [-0.0017, +0.0038]$, $p = 0.347$	$+0.0011 [-0.0050, +0.0073]$, $p = 0.636$
Without DBSCAN, negation retained	12,192	3,049	0.8094 ± 0.0081	0.6440 ± 0.0055

DISCUSSIONS

The central result is that removing DBSCAN-flagged documents from the training data did not improve the classifier on the fixed held-out test set. The paired difference of $+0.0010$ in accuracy carries a confidence interval that spans zero, so the finding is an absence of evidence of benefit rather than a measured harm; the deterioration reported by the earlier, unpaired version of this experiment does not survive a matched comparison. The finding is worth reporting, because filtering adds computation and discards usable training data, so a procedure that does not demonstrably help should not be adopted by default. Three candidate explanations follow, each stated with the evidence in this study that bears on it.

The first concerns what DBSCAN actually identified, and it is now supported by direct evidence rather than offered as a conjecture. In a TF-IDF representation of short reviews, a document falls into a low-density neighbourhood when it uses uncommon vocabulary or is too short to share terms with its neighbours, not when its opinion is extreme or its label is wrong, which is the distinction drawn by Frenay and Verleysen (2014). Of the forty flagged documents read in Table 6, not one was spam, a string of repeated characters, or a comment without opinion. Thirty-nine were ordinary short reviews such as Terima kasih Aplikasi nya keren dan amanah, and the fortieth was a valid text whose rating contradicted it. The filter is therefore discarding usable training data, and the label DBSCAN attaches to these documents, noise, does not describe them.

The second concerns the geometry of the feature space. Distances in the hundred-component decomposition of a sparse TF-IDF matrix concentrate within a narrow range, so the contrast between dense and sparse neighbourhoods is weak; the k-distance curve in Fig. 1 rises gradually rather than showing a sharp knee, which is why eps had to be fixed by a percentile rule. If this account is correct, no setting of the filter should produce a clear improvement, and Table 7 confirms the prediction: a twentyfold change in the number of removed documents moves accuracy by less than a percentage point.

The third explanation, that filtering interacts with class imbalance by stripping the already-scarce neutral class, is contradicted by the evidence and is withdrawn. Table 5 shows that of the 338 flagged documents, 307 are positive, 25 negative and only 6 neutral. The filter removed 3.98 percent of the positive class but only 0.84 percent of the neutral class, so the minority class was the least affected, not the most. The pattern is explained by the geometry rather than by the labels: the positive class is the largest and most repetitive, so its densest neighbourhoods are tight and the short, enthusiastic reviews that sit just outside them fall below the density threshold. Class imbalance remains the binding constraint on macro-averaged performance, as the neutral F1-score of 0.26 to 0.28 shows, but it operates through the classifier rather than through the filter.

Two further points should be stated. First, the protocol is paired, and no test document is removed, so the difference between the two rows of Table 9 is not confounded by a change in the evaluation sample. This corrects an earlier version of this experiment in which filtering preceded the split; there the scenarios were scored on test sets of 3,049 and 2,961 documents, and the apparent drop in accuracy from 0.786 to 0.778 was an artefact of that difference. Second, an incidental finding deserves more attention than the filtering stage. The reference preprocessing destroys negation, because the slang dictionary maps *gak*, *tdk* and *tak* onto *tidak* while *tidak* and *belum* are Sastrawi stopwords, so *lama dan tdk gratis* becomes *lama gratis*. Retaining the negation particles raised accuracy from 0.8023 to 0.8094 and macro F1-score from 0.6390 to 0.6440, several times larger than anything the filter produced.

Placed alongside the studies in Table 1, the baseline accuracy obtained here is broadly consistent with the range reported for support vector machines on Indonesian application reviews, while remaining below figures reported for balanced binary datasets, which the three-class formulation and the retention of the natural class distribution explain. The comparison is in any case indirect, because those studies use different corpora and preprocessing.

Beyond the modelling question, the class vocabularies carry practical value for the provider: *biaya* admin, *cepat*, and *bantu* in the positive class confirm that free transfers and speed drive satisfaction, while *lama*, *lambat*, *refund*, and *layan langgan* in the negative class identify delayed transactions, refund handling, and customer service as the areas where intervention would most reduce negative sentiment.

CONCLUSION

This study examined whether removing density-based outliers from the training data improves the classification of Flip reviews. From 20,000 collected records, 15,318 remained after duplicate removal and 15,241 after preprocessing, and these were split once into stratified training and test partitions before any component was fitted, so that TF-IDF, the truncated decomposition and DBSCAN were estimated on the 12,192 training documents alone and both scenarios were scored on the same 3,049 held-out documents. Because the k-distance curve showed no pronounced knee, eps was fixed at the ninetieth percentile of that distribution. Averaged over five stratified splits, accuracy was 0.8023 without filtering and 0.8033 with it, a paired difference of +0.0010 with a 95 percent confidence interval of [-0.0017, +0.0038] and $p = 0.347$; macro F1-score moved from 0.6390 to 0.6402, interval [-0.0050, +0.0073]. Both intervals contain zero, and no combination in the nine-point parameter grid improved on the baseline, so the study found no evidence that density-based filtering helps this classifier. Because filtering costs computation and discards usable training data, it should not be adopted here by default.

Two findings qualify that conclusion. The documents the filter discards are not noise: of forty inspected, none was spam or gibberish, and 91 percent of all flagged documents belonged to the majority positive class rather than to the scarce neutral class. The binding constraint on macro-averaged performance is the neutral class itself, whose F1-score stays near 0.27 in both scenarios because its precision is below 0.21 even with balanced class weights. An incidental result is larger than either:

retaining negation particles, which the reference pipeline removes, raised accuracy to 0.8094 and macro F1-score to 0.6440.

Several limitations should be acknowledged. The labels are derived from star ratings rather than from manual annotation, and the validation against two annotators described in the method had not been completed at the time of submission, so the classifier is best described as predicting the rating category from the review text. Performance was estimated from five stratified splits of one corpus rather than from independent corpora. The sensitivity analysis covers eps and minPts but holds the representation, the distance metric and the classifier fixed, and resampling was not tested alongside class weighting. The manual inspection covers forty documents and was performed by the authors rather than by independent annotators.

Future work should proceed in three directions. Alternative anomaly detection methods, including the local outlier factor and the isolation forest, and label-oriented filtering such as the edited nearest neighbour rule, should be compared against DBSCAN under the same paired protocol and the same fixed test set (Ratih et al., 2026). Methods that address the minority class directly, such as synthetic minority oversampling used alongside class weights, should be evaluated against the filtering stage rather than in place of it, so the two interventions can be ranked (He & Garcia, 2009). Finally, contextual representations from Indonesian pretrained transformer models (Devlin et al., 2019) may separate ambivalent reviews from mildly polar opinion better than lexical features.

ACKNOWLEDGMENT

The authors gratefully acknowledge the institutional support of Sekolah Tinggi Manajemen Informatika dan Komputer Jayakarta, Politeknik Negeri Cilacap, and Piksi Input Polytechnic, without which this study could not have been carried out: the research time granted by the leadership of each institution, the laboratory and computing facilities and the library and journal subscriptions provided by the Department of Informatics Engineering, the Department of Computer and Business, and the Cybersecurity Engineering Study Program, and the discussion of earlier drafts offered by colleagues at all three campuses. The authors also thank the anonymous reviewers, whose detailed criticism reshaped the experimental design reported here. This research received no specific grant from any funding agency.

REFERENCES

- Aggarwal, C. C. (2017). *Outlier analysis* (2nd ed.). Springer. <https://doi.org/10.1007/978-3-319-47578-3>
- Antara News. (2022, July 27). *Flip masuk 10 besar aplikasi pembentuk generasi versi Google Play*. <https://www.antaranews.com/berita/3020701/flip-masuk-10-besar-aplikasi-pembentuk-generasi-versi-google-play>
- Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). LOF: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data* (pp. 93–104). <https://doi.org/10.1145/342009.335388>
- Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3), 1–58. <https://doi.org/10.1145/1541880.1541882>
- Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297. <https://doi.org/10.1007/BF00994018>
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019* (pp. 4171–4186). <https://doi.org/10.18653/v1/N19-1423>
- Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)* (pp. 226–231).
- Frénay, B., & Verleysen, M. (2014). Classification in the presence of label noise: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5), 845–869. <https://doi.org/10.1109/TNNLS.2013.2292894>

- Hamid, R. B., Andriansyah, C., Sensuse, D. I., Lusa, S., Elisabeth, D., & Safitri, N. (2025). Sentiment analysis and topic modeling for discovering knowledge in Indonesian mobile government applications. *Jurnal Teknik Informatika (JUTIF)*, 6(5), 3188–3203. <https://doi.org/10.52436/1.jutif.2025.6.6.4991>
- He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9), 1263–1284. <https://doi.org/10.1109/TKDE.2008.239>
- Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning (ECML-98)* (pp. 137–142). <https://doi.org/10.1007/BFb0026683>
- Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1), 1–167. <https://doi.org/10.2200/S00416ED1V01Y201204HLT016>
- Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *Proceedings of the Eighth IEEE International Conference on Data Mining* (pp. 413–422). <https://doi.org/10.1109/ICDM.2008.17>
- Madjid, M. F., Ratnawati, D. E., & Rahayudi, B. (2023). Sentiment analysis on app reviews using support vector machine and naïve Bayes classification. *Sinkron: Jurnal dan Penelitian Teknik Informatika*, 7(1), 556–562. <https://doi.org/10.33395/sinkron.v8i1.12161>
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press. <https://doi.org/10.1017/CBO9780511809071>
- Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*, 5(4), 1093–1113. <https://doi.org/10.1016/j.asej.2014.04.011>
- Mustakim, H., & Priyanta, S. (2022). Aspect-based sentiment analysis of KAI Access reviews using NBC and SVM. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, 16(2), 113–124. <https://doi.org/10.22146/ijccs.68903>
- Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1–2), 1–135. <https://doi.org/10.1561/15000000011>
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- PT Fliptech Lentera Inspirasi Pertiwi. (2026). *Flip: Kirim, bayar, hemat lebih banyak*. <https://flip.id>
- Purwanto, J., Mahardika, F., & Nugroho, A. (2026). Automatic Dewey decimal classification of Indonesian book metadata using IndoBERT with weighted loss and context enhancement. *Journal of Artificial Intelligence and Technology Information (JAITI)*, 4(2), 171–181. <https://doi.org/10.58602/jaiti.v4i2.258>
- Purwanto, R., Afif, R., Mahardika, F., Prasetyanti, D. N., Vikasari, C., & Listyaningrum, R. (2026). User sentiment toward Halodoc psychological consultation services using RNN and LSTM analysis. *Journal of Innovation Information Technology and Application (JINITA)*, 8(1), 301–315. <https://doi.org/10.35970/jinita.v8i1.3288>
- Ratih, R., Yuniarto, A. H. P., Ichwan, H., Mahardika, F., & Ripai, R. (2026). Performance evaluation of random forest and isolation forest algorithms for detecting anomalies in SIAKAD server log data. *Sinkron: Jurnal dan Penelitian Teknik Informatika*, 10(2), 1220–1230.
- Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5), 513–523. [https://doi.org/10.1016/0306-4573\(88\)90021-0](https://doi.org/10.1016/0306-4573(88)90021-0)
- Schubert, E., Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (2017). DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN. *ACM Transactions on Database Systems*, 42(3), 1–21. <https://doi.org/10.1145/3068335>
- Setyani, T., Sari, K., Heidy, H. N., & Suryono, R. R. (2026). Analisis sentimen pengguna aplikasi Jamsostek Mobile berdasarkan ulasan Google Play Store menggunakan algoritma support vector machine dan naïve Bayes. *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, 6(1), 373–384. <https://doi.org/10.57152/malcom.v6i1.2526>

-
- Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4), 427–437.
<https://doi.org/10.1016/j.ipm.2009.03.002>
- Wilson, D. L. (1972). Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-2(3), 408–421.
<https://doi.org/10.1109/TSMC.1972.4309137>