Journal of Emerging Information Systems and Business Intelligence
ISSN: 2774-3993
Journal homepage: https://ejournal.unesa.ac.id/index.php/JEISBI/

Web-based Profanity Detection Using a Combination of Lexicon and Support Vector Machine

Ainandita Riwipapusa[1], Wiyli Yustanti[2]

[1,2]*Universitas Negeri Surabaya, Surabaya, Indonesia*

ainandita.21029@mhs.unesa.ac.id, *wiyliyustanti@unesa.ac.id*

ABSTRACT

Advances in information and communication technology, particularly the internet and social media, have made it easier for people to express their opinions openly, but have also increased the potential for the spread of profanity and hate speech. This study proposes a web-based profanity detection solution by combining lexicon-based methods and Support Vector Machine (SVM). The Knowledge Discovery in Database (KDD) process was implemented for data extraction and analysis, starting from Twitter data collection, preprocessing (cleaning, case folding, tokenizing, stemming), transformation using TF-IDF, to manual labeling. The SVM model was trained using a 3-fold cross-validation scheme, and evaluation was conducted using a classification report and confusion matrix. The results of the study showed a model accuracy of 93% on the test data with an average F1-score of 0.93, as well as optimal performance in detecting sentences categorized as profanity. The developed web application prototype successfully ran all profanity word detection and sensing features automatically, as proven by the black box testing results. The analysis test also ran smoothly, with a test using 10 sentences containing profanity words achieving 100% accuracy, and a test using 10 sentences without profanity words achieving 95% accuracy. This system is expected to contribute to creating a more positive digital space through adaptive and accurate profanity word detection.

Keyword: Super Vector Machine (SVM), Lexicon-Based, Detection, Profanity, Knowledge Discovery in Database (KDD), Prototype

Article Info:
Article history:
Received July 15, 2025
Revised August 21, 2025
Accepted September 10, 2025

Corresponding Author
Wiyli Yustanti
Universitas Negeri Surabaya, Surabaya, Indonesia
wiyliyustanti@unesa.ac.id

1. INTRODUCTION

Advances in information and communication technology, particularly the internet and social media, have brought about major changes in the way people interact. This has enabled everyone to freely express their opinions on various digital platforms. However, this freedom has also given rise to new challenges, as the use of profanity and hate speech is often found [1]. Therefore, a profanity detection system is greatly needed.

Automatic detection of profanity is very important for maintaining a healthy digital ecosystem. Various approaches have been developed to address this issue, ranging from lexicon-based approaches to the use of machine learning techniques. Lexicon-based approaches can simplify the labeling process by using existing lists of profanity. However, if

the lexicon does not include new words, detection accuracy may decrease, and it may also be less effective in handling complex contexts or ambiguous phrases [2].

Previous research implemented SVM for detecting negative comments on the Twitter platform with 254 comment data, achieving an accuracy of 88%, precision of 100%, recall of 50%, and an F1 score of 67% [4]. Meanwhile, another study compared the SVM method with a lexicon-based method in sentiment analysis using a dataset of 4,000 data points. The accuracy of the SVM model was found to be 98.5%, while the lexicon-based method performed poorly in analysis, achieving an accuracy of 78.43% [5]. This indicates that SVM is more effective in handling the complexity of language and context in sentiment analysis compared to lexicon-based methods.

Based on this background, a relevant and innovative solution was developed by combining the Support Vector Machine (SVM) method and a lexicon-based approach. This application is expected to provide accurate and adaptive detection results for language dynamics on social media, as well as contribute to efforts to create a more positive digital space.

2. METHODS

The following is the research flow used for this study. This study uses the Knowledge Discovery in Database (KDD) method to build an SVM model used for sentence sentiment and the Prototype method for the application development process. The following research flow framework is illustrated in Figure 1.

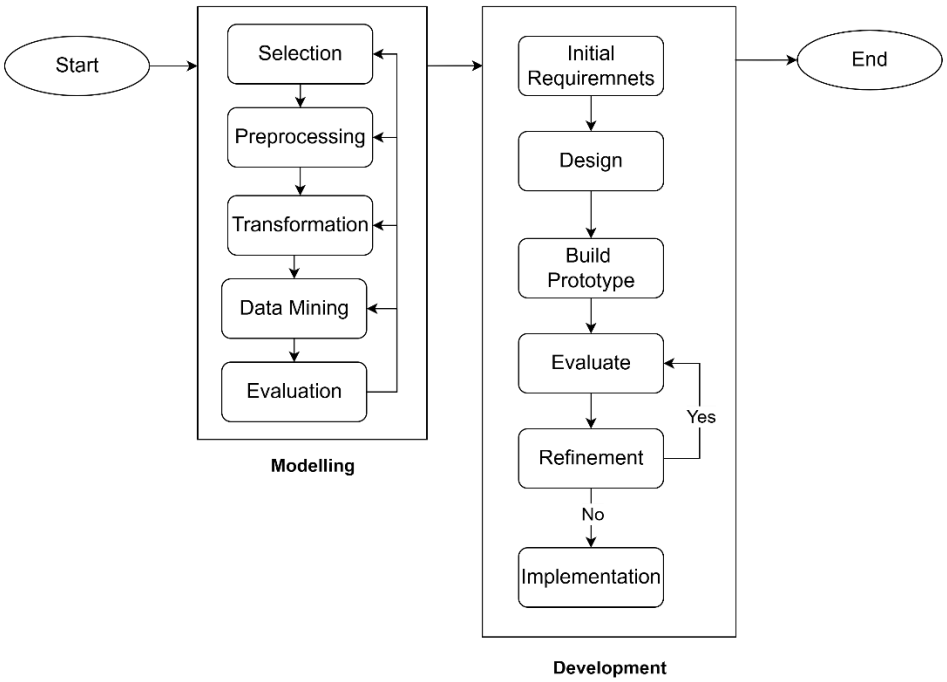

Figure 1. Research Framework

In the initial step, an SVM model was constructed by employing the Knowledge Discovery in Database (KDD) methodology. The KDD process aims to extract and analyze large datasets to uncover valuable information and knowledge through data mining techniques [6]. KDD is not just a data mining process, but also a systematic series of steps to convert raw data into valuable knowledge [7]. There are five processes involved in KDD, which are described as follows:

A. Selection

At this stage, data collection was carried out to obtain data that was relevant and appropriate to the model requirements. The data used in this study was obtained from Twitter using crawling techniques. The data was collected using several specific keywords that had the potential to contain profanity or negative opinions in Indonesian. The crawling results yielded 1,502 sentences.

B. Preprocessing

Preprocessing is done to prepare the data for the next stage. There are four common steps in the preprocessing process: case folding, tokenizing, filtering, and stemming [8]. At this stage, the dataset, which previously contained many unnecessary characters and punctuation marks, is processed. This cleans up the data and makes it ready for analysis or modeling in the next stage.

The first step is the cleaning process. This process is carried out to clean the dataset of characters that are not needed in the classification process. Cleaning is done by removing punctuation marks, emoticons, foreign characters, URLs, and so on. This is done to convert the text format to a standard form so that it is easier to process. The cleaning process can be seen in Table 1.

Table 1. Cleaning

Process Input	Process Ouput
@riz_esp Anak anjing dan anak babi tu anak2 dia juga ke? Tanya je jangan marah	Anak anjing dan anak babi tu anak dia juga ke Tanya je jangan marah

Next, case folding is performed, which involves changing sentences or words from the dataset. Words or sentences that were previously in uppercase will be changed to lowercase. This is done to reduce unnecessary word variation. The case folding process can be seen in Table 2.

Table 2. Case Folding

Process Input	Process Ouput
Anak anjing dan anak babi tu anak dia juga ke Tanya je jangan marah	anak anjing dan anak babi tu anak dia juga ke tanya je jangan marah

The next stage is the tokenizing process. This process is carried out by separating sentences into words or tokens. The purpose of this process is to facilitate statistical analysis of the text, such as calculating the frequency of word occurrence, as well as preparing data for further processing stages. The tokenizing process can be seen in Table 3.

Table 3. Tokenizing

Process Input	Process Ouput
anak anjing dan anak babi tu anak dia juga ke tanya je jangan marah	‘anak’, ‘anjing’, ‘dan’, ‘anak’, ‘babi’ ‘tu’, ‘anak’, ‘dia’,

	‘juga’, ‘ke’, ‘tanya’, ‘je’, ‘jangan’, ‘marah’

The final stage is the stemming process. This process is used to convert words into their basic form. The purpose of this process is to standardize various forms of words so that they are recognized as the same entity by the model and are not treated as different words. The stemming process can be seen in Table 4.

Table 4. Stemming

Process Input	Process Ouput
anak anjing dan anak babi tu anak	anak anjing dan anak babi anak
dia juga ke tanya je jangan marah	dia juga tanya jangan marah

C. Transformation

During the data transformation stage, text data is converted into numerical representations that can be processed by machine learning algorithms. Previously, the collected text data underwent a manual labeling stage, which involved manually labeling each sentence to classify it as profanity or non-profanity.

After labeling, the TF-IDF technique was used, which is a combination of two words, namely Term Frequency and Inverse Document Frequency [9]. This technique calculates the weight of words based on their relative frequency in the corpus, so that distinctive words are given a higher weight. The result is a numerical vector feature matrix that is used as input for SVM classification.

D. Data Mining

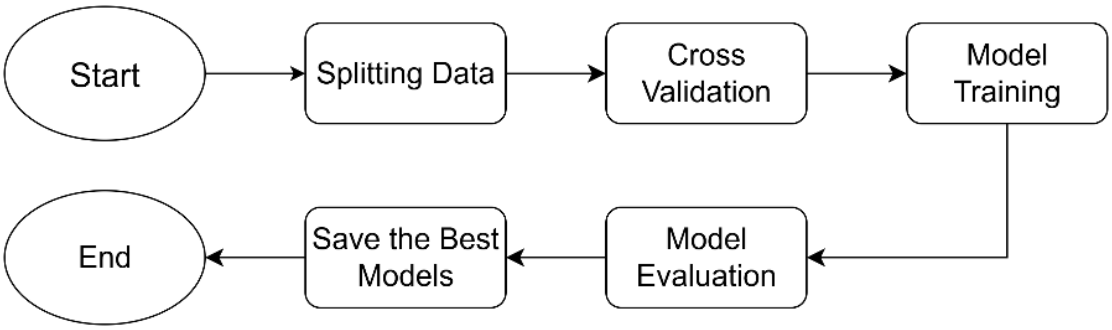

Figure 2. Data Mining Flow

The data mining process flow can be seen in Figure 2. This stage will build a model using the Super Vector Machine (SVM) algorithm. First, the data is split into two sets: the first set for training and validation data, and the second set for test data. The data is split in an 80:20 ratio, where 80% of the training and validation data is used for training and cross-validation, and 20% of the test data is used for final evaluation after the best model is identified.

Next, cross-validation is performed using StratifiedKFold with a number of 3-fold. The SVM model will train the data three times with different data combinations. Each fold will produce accuracy from the training results. The values obtained are used to evaluate which fold has the best model performance. After that, the model is saved for use in the next process.

E. Evaluation

In the final evaluation stage, the best SVM model selected during the cross-validation process was tested on test data that had been previously separated from the entire dataset. The test data used was data that had never been seen by the model before. The evaluation was carried out using classification report metrics that included precision, recall, and F1-Score for each class, as well as overall accuracy. Additionally, an evaluation was conducted using a confusion matrix to facilitate the interpretation of the distribution of correct and incorrect predictions for each class.

The next step is to develop the interface using the prototype method. Prototyping is used to build initial models that can be evaluated and tested with users to clarify user needs and expectations for the system to be developed [10]. This method has the advantage of allowing dynamic adaptation to user needs. The following are the stages of the prototype method [11]. There are six processes involved in this stage, which are described as follows:

A. Requirements Analysis

This stage includes analyzing the requirements to determine the components and resources needed to build a profanity detection system. Some of the analyses are as follows.

1) User Requirements Analysis
   The users of this application are individuals or institutions who want to filter text content containing profanity. Therefore, this system is designed to have a simple and easy-to-use interface for entering text and displaying detection results.
2) Hardware Requirements Analysis
   The hardware requirements for building the application are an 11th Gen Intel(R) Core(TM) processor, 8GB RAM, a keyboard, and a mouse.
3) Software Requirements Analysis
   The software needed to build the application includes Windows 11 operating system, Google Chrome browser, Google Colaboratory, Sublime Text, and Draw.io.

B. Design

The design serves as a reference in building an initial prototype that meets user needs and application objectives. The interface design is carried out to provide an initial illustration of the model that will be developed further.

1) Use Case Diagram

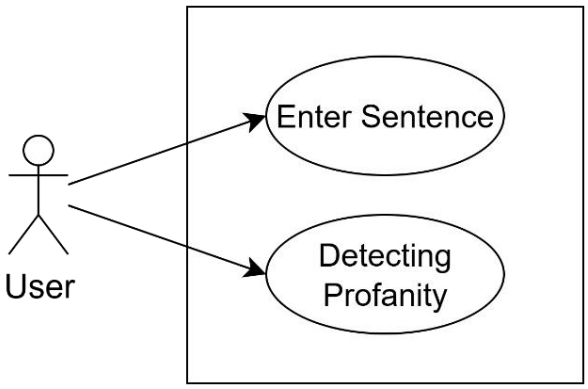

Figure 3. Use Case Diagram

The use case design that illustrates the functional relationship between user input and the system's analytical capabilities is shown in Figure 3. In this design, the usage presented in this study illustrates the interaction flow in which users can enter sentences to be analyzed by

the system. Next, the system processes these sentences using detection features to identify inappropriate or irrelevant content.

2) Activity Diagram

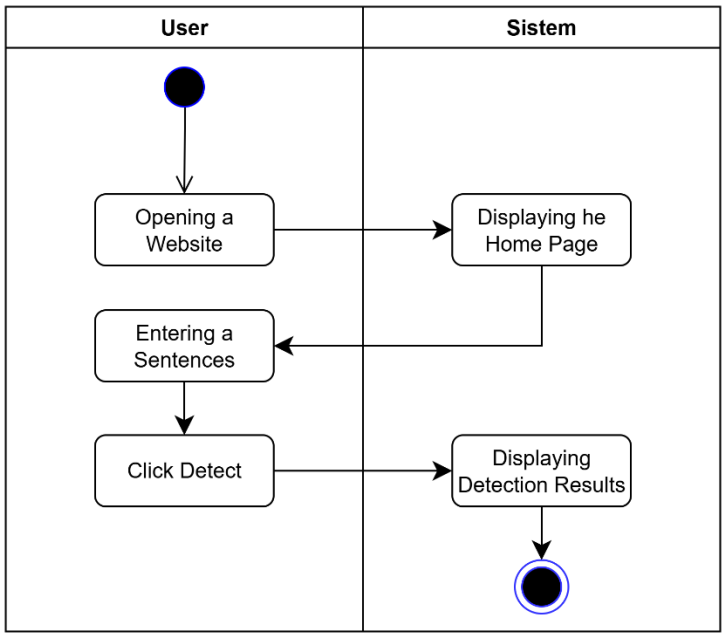

Figure 4. Activity Diagram

The activity diagram design can be seen in Figure 4. The activity diagram illustrates the flow of activities that users can perform in the application. This process begins when users access the website and enter a sentence in the field provided.

3) Sequence Diagram

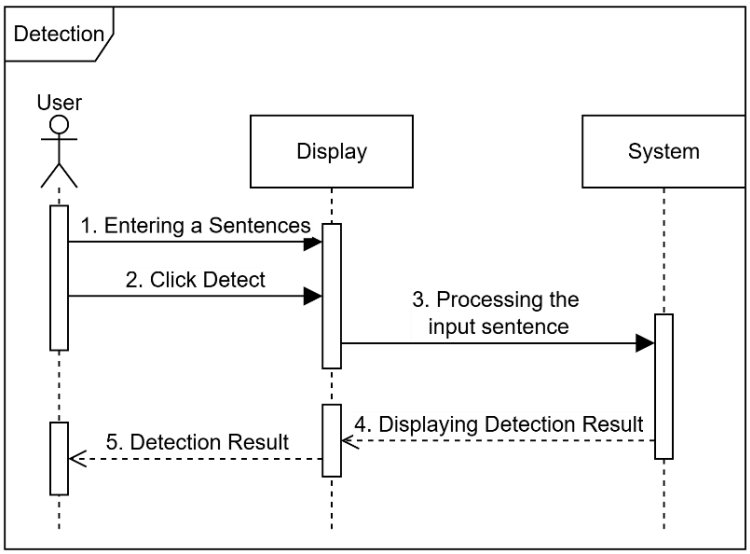

Figure 5. Sequence Diagram

The sequence diagram design can be seen in Figure 5. The sequence diagram explains that the process begins when the user enters a sentence. This input is then sent to the system for processing. The system will process the input using a pre-designed detection algorithm.

After that, the detection results are returned to the application interface and displayed to the user.

4) Interface Design

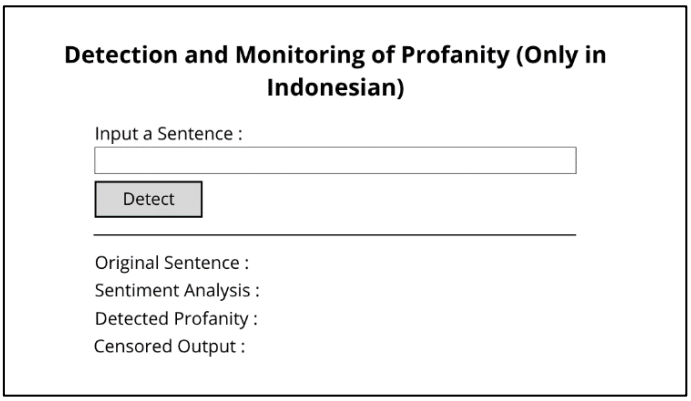

Figure 6. Interface Design

The initial interface design can be seen in Figure 6. This is a preliminary design for the interface that will be built. Users can input the sentence they want to detect in the “Input a Sentence” column. Then, the inputted sentence can be detected by clicking the “Detect” button. As a result, the sentiment analysis, detected profanity, and censored output of the detected sentence will appear.

C. Build Prototype

At this stage, system coding is carried out to build the application interface. This stage involves code development, feature integration, and further testing [11]. This stage is the output of the requirements determination process in the initial stage [12].

This design was developed using the Python programming language with the built-in Jinja Template from Flask as the backend framework and HTML as the website display. In this implementation, sentence classification is performed using the SVM model built in the previous stage, while word classification uses a lexicon-based approach.

D. Evaluation

This evaluation stage serves to identify potential problems in the interface display that could interfere with user performance in running the system [13]. The evaluation was carried out using 20 sentences that were input into the application. Ten sentences contained profanity, and ten sentences did not contain profanity. The evaluation was carried out using the Black Box Testing technique and analysis testing.

E. Refinement

This stage is carried out after the application has been evaluated. Based on the evaluation, the prototype improvement stage is carried out [14]. Improvements are made based on input and suggestions provided by users until the prototype meets their needs and expectations.

F. Implementation

The final stage is the implementation process. This includes final coding, testing, user training, and initial monitoring to ensure that the system runs according to requirements, so that users can make optimal use of the solution. Implementation is carried out by developing

the system based on a validated prototype to accelerate development and minimize the risk of failure [15].

3. RESULTS AND DISCUSSION

3.1 PROFANITY CLASSIFICATION

This section shows the results of the SVM model in detecting profanity. The architectural parameters used in this model can be seen in Table 5.

Table 5. Architecture Parameters

Parameter	Setting	Standard
kernel	'linear'	
c	1.0	'rbf'
gamma	'scale'	'scale'
degree	3	3
coef0	0.0	0.0
probability	False	False
shrinking	True	True
class_weight	None	None
random_state	42	None

The model was trained using an 80:20 data split and 3-fold cross-validation of the total data consisting of 1,502 sentences. The data used was divided into two sets, where the first set was the training and validation data consisting of 1,202 sentences, and the second set was the test data consisting of 300 sentences. The modeling results can be seen in Table 6.

Table 6. Model Accuracy Results

Fold	Accuracy
1	93,75%
2	92,75%
3	91,50%

The best accuracy result was obtained by fold 1 with an accuracy value of 93.75%. Then, the model from this fold was saved for use in building the interface. An evaluation of this model was also carried out using the previously saved test data. The evaluation was carried out using a classification report and confusion matrix. The evaluation results using the classification report can be seen in Figure 7.

```
Classification Report:
                   precision    recall  f1-score   support

Non-Profanity (0)       0.95      0.85      0.90       179
    Profanity (1)       0.89      0.96      0.93       221

         accuracy                           0.92       400
        macro avg       0.92      0.91      0.91       400
     weighted avg       0.92      0.92      0.91       400
```

Figure 7. Classification Report

Based on the evaluation results, the model achieved an accuracy of 93%, with a macro and weighted f1-score of 0.93, respectively. In class 0 or non-profanity, precision was 0.97 and recall was 0.87, while in class 1 or profanity, precision reached 0.91 with recall of 0.98.

These results indicate that the model performs well and is highly effective in detecting class 1, although it is slightly less sensitive to class 0. The results of the evaluation using the confusion matrix can be seen in Figure 8.

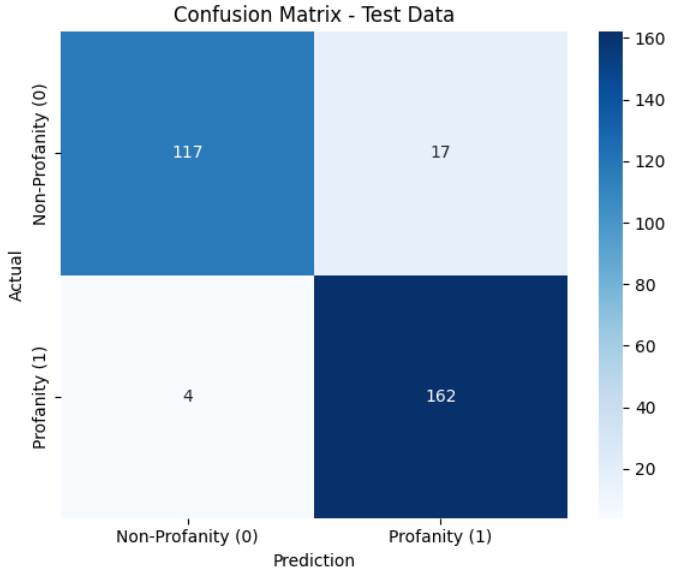

Figure 8. Confusion Matrix

Based on the evaluation results using the confusion matrix, the model was able to classify the data quite well. Of the total 134 actual data points in the ‘not profanity’ class, 117 were correctly classified, while 17 were incorrectly classified as ‘profanity’. Meanwhile, out of the 166 actual data points in the ‘profanity’ class, 162 were correctly classified, and only 4 were incorrectly classified as ‘not profanity’.

These results show that the model has excellent ability to recognize data containing profanity, with a low error rate. In addition, the classification error rate for the “non-profanity” class is also relatively small and still within reasonable limits.

3.2 DEVELOMPENT

This section presents the results of the application interface development that has been adapted to the design formulated in the previous stage.

A. Interface

Figure 9. Home Page

As can be seen on the home page interface in Figure 9, users can enter the sentences they want to detect. After writing the sentence, they can click the “Detect” button, and the sentence will be detected by the system.

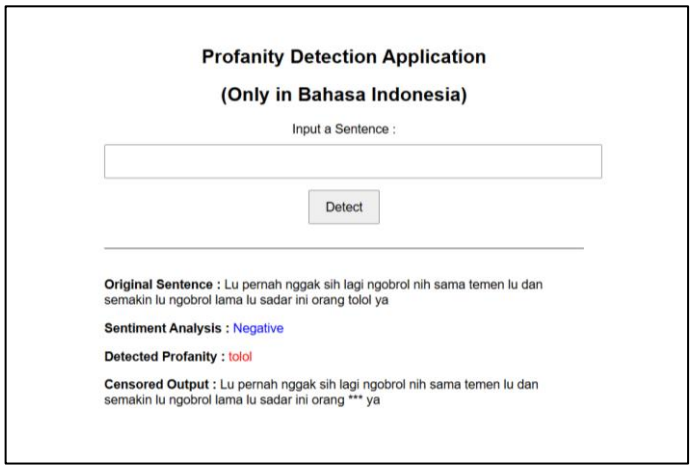

Figure 10. Detection Ouput Interface

The detection results can be seen in Figure 10. The detection results display the sentiment analysis of the entered sentence, which can be either ‘Negative’ or ‘Non-Negative’. This section also displays words that are detected as profanity. The censored output section displays sentences with censorship on detected profanity.

B. Black Box Testing

Testing was conducted using the black box testing method. This test will examine the system's functions and focus on the system's inputs and outputs performed by users. The results of black box testing can be seen in Table 7.

Table 7. Black Box Testing

Test Scenario	Description	Expected Results	Result
Users can input sentences	The sentences is input by entering it into the provided text field	The system receives a sentences input	✓
Users can detect sentences	The users triggers the detection process by clicking the ‘Detect’ button	Display detection results including sentiment analysis, detected profanities, and the sentence with censored words	✓

The results of black box testing show that all features work well in all test scenarios, marked with a ‘✓’ in the results section.

C. Analytical Test

Next, an analysis test was conducted to further test the system's ability to detect profanity. This test was conducted by entering 10 sentences containing profanity and 10 sentences without profanity. The results of the system analysis test can be seen in Table 8.

Table 8. Analytical Test

Label	Successful	Accuracy
Profanity	10/10	100%
Non-Profanity	19/20	95%

The test results show that sentences containing profanity were detected correctly, resulting in an accuracy rate of 100%. Meanwhile, sentences that did not contain profanity had one detection error, resulting in an accuracy rate of 95%.

CONCLUSION

Based on the research results, it can be concluded that the developed SVM model successfully detected profanity with good performance, as shown by an accuracy of 93.75% on the training data and 93% on the test data with an average F1 score of 0.93. Evaluation using the confusion matrix indicates a low classification error rate, particularly in identifying sentences falling under the profanity category.

The implementation of the model on the web application also runs optimally, as evidenced by all features meeting expectations in black box testing. Additionally, analysis testing with 10 sentences containing profanity and 10 without profanity yielded results where sentences containing profanity achieved 100% accuracy, while sentences without profanity achieved 95% accuracy. This indicates that the system built is effective in automatically detecting and censoring profanity.

REFERENCES

[1] R. Deswandi Yahya, S. Adi Wibowo, and N. Vendyansyah, "Analisis Sentimen Untuk Deteksi Ujaran Kebencian Pada Media Sosial Terkait Pemilu 2024 Menggunakan Metode Support Vector Machine," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 8, no. 2, pp. 1182–1189, 2024, doi: 10.36040/jati.v8i2.9076.

[2] R. Darman, "ANALISIS SENTIMEN RESPONS TWITTER TERHADAP PERSYARATAN BADAN PENYELENGGARA JAMINAN SOSIAL (BPJS) DI KANTOR PERTANAHAN," *J. Widya Bhumi*, vol. 3, no. 2, pp. 113–136, 2023.

[3] P. Sunarko, A. Bijaksana, P. Negara, and R. Septiriana, "Perbandingan Klasifikasi Algoritma Support vector machine dan Naïve Bayes Menggunakan Labeling VADER dan Lexicon based pada Tweets Bahasa Indonesia dan Bahasa Inggris Comparison of Support vector machine and Naïve Bayes Classification Algorithms Using VAD," vol. 03, no. 1, pp. 9–19, 2024, doi: 10.26418/juara.v3i1.86468.

[4] A. R. Iqbal and Y. Miftahuddin, "Implementasi SVM Untuk Deteksi Komentar Negatif Berbahasa Indonesia di Twitter," *Fti*, vol. X, no. X, 2022, [Online]. Available: https://eproceeding.itenas.ac.id/index.php/fti/article/view/966%0Ahttps://eproceeding.itenas.ac.id/index.php/fti/article/download/966/942

[5] M. R. A. Yudianto, A. Rahim, P. Sukmasetya, and R. A. Hasani, "Perbandingan Metode Support Vector Machine Dengan Metode Lexicon Dalam Analisis Sentimen Bahasa Indonesia," *J. Teknol. Inf.*, vol. 6, no. 1, pp. 7–13, 2022, [Online]. Available: https://github.com/fajri91/InSet.

[6] I. K. J. Arta, G. Indrawan, and G. Rasben Dantes, "Data Mining Rekomemdasi Calon

Mahasiswa Berprestasi di STMIK Denpasar Menggunakan Metode Technique For Other Reference By Similarity to Ideal Solution," *J. Ilmu Komput. Indones.*, vol. 4, no. 1, pp. 11–21, 2019.

[7] C. Zhang and J. Han, *Data Mining and Knowledge Discovery*, vol. 10, no. 11. 2021. doi: 10.1007/978-981-15-8983-6_42.

[8] L. Hermawan and M. B. Ismiati, "Pembelajaran Text Preprocessing berbasis Simulator Untuk Mata Kuliah Information Retrieval," vol. 17, no. 2, pp. 188–199, 2020.

[9] R. Romindo, J. J. Pangaribuan, and O. P. Barus, "Implementasi Algoritma Tf-Idf Dan Support Vector Machine Terhadap Analisis Pendeteksi Komentar Cyberbullying Di Media Sosial Tiktok," *Device*, vol. 13, no. 1, pp. 124–134, 2023, doi: 10.32699/device.v13i1.5260.

[10] D. A. N. Wulandari, A. A. H. Bahar, M. G. Arfananda, and H. Apriyani, "Prototyping Model in Information System Development of Al-Ruhamaa' Bogor Yatim Center Foundation," *Pilar Nusa Mandiri J. Comput. Inf. Syst.*, vol. 17, no. 2, pp. 127–136, 2021, [Online]. Available: http://ejournal.nusamandiri.ac.id/index.php/pilar/article/view/2375

[11] P. Kustanto, B. K. Ramadhan, and A. Noe, "Penerapan Metode Prototype dalam Perancangan Media Pembelajaran Interaktif," vol. 5, no. 1, pp. 83–94, 2025.

[12] S. Supiyandi, C. Rizal, and B. Fachri, "Implementasi Model Prototyping Dalam Perancangan Sistem Informasi Desa," *Resolusi Rekayasa Tek. …*, vol. 3, no. 3, pp. 211–216, 2022, [Online]. Available: http://djournals.com/resolusi/article/view/611%0Ahttps://djournals.com/resolusi/article/download/611/396

[13] H. Akmal, I. P. Gede, and H. Suputra, "Evaluasi UI pada Prototype Aplikasi ' WeCare ' Menggunakan Metode SUS (System Usability Scale)," vol. 2, no. November, pp. 131–138, 2023.

[14] A. S. Khairi and Putrawan, "Perancangan Prototype Cloud Computing Dalam Menyimpan Tugas Dan Pembelajaran Mahasiswa Di Kelas Menggunakan Owncloud," *J. Komput. Teknol. Inf. dan Sist. Inf.*, vol. 1, no. 3, pp. 161–169, 2023, doi: 10.62712/juktisi.v1i3.32.

[15] A. Fikriyya and R. T. Dirgahayu, "Implementasi Prototyping dalam Perancangan Sistem Informasi Sekolah Desa Pendar Foundation Yogyakarta," *J. UII Autom.*, vol. 1, no. 2, pp. 1–9, 2020.