Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 5, **Number** 1, May 2025
https://doi.org/10.47709/brilliance.v5i1.6580

Sentiment Analysis of Mental Health Using Support Vector Machine (SVM) with FastAPI Implementation

Maulyanda[1*], Sri Azizah Nazhifah[2]
[1,2]Universitas Syiah Kuala, Indonesia
[1]maulyanda@usk.ac.id, [2]sriazizah07@usk.ac.id

open access

***Corresponding Author**

Article History:
Submitted: 05-07-2025
Accepted: 20-07-2025
Published: 24-07-2025

Keywords:
Support Vector Machine (SVM); Sentiment Analysis; Mental Health; FastAPI.

Brilliance: Research of Artificial Intelligence is licensed under a Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0).

ABSTRACT

Mental health is a vital aspect that contributes significantly to an individual's productivity, daily activity, and overall quality of life. With the increasing prevalence of mental health issues, early detection and analysis are essential. This study aims to identify mental health conditions using a machine learning approach, specifically the Support Vector Machine (SVM) algorithm. The dataset used consists of 53,043 text-based statements that are classified into seven distinct categories of mental conditions: normal, depression, suicide, anxiety, bipolar, stress, and personality disorders. The preprocessing steps applied to the dataset include text cleaning, tokenization, stopword removal, and lemmatization to standardize the textual input. Following this, 80% of the data is allocated for training the model, while the remaining 20% is reserved for testing purposes. The SVM algorithm is utilized to build a predictive model capable of classifying mental conditions based on text input. Furthermore, this model is deployed through an application interface using the FastAPI framework, enabling integration with digital platforms. The results indicate that the model achieves an accuracy of 79%, a recall of 77%, and an F1-score of 73%. These findings suggest that SVM is a capable and efficient method for analyzing and detecting various mental health conditions. This approach supports early intervention and offers practical implications for digital mental health screening tools.

INTRODUCTION

As with physical health, the mental health is also critical component for maintaining an individual's overall health. A mentally health individual is able to recognize their own potential, overcome the pressure of life and work, manage daily life, and contribute to positively social activities, especially in teenager, the mental is crucial issue that have to be detected in early stage (Supini, Gandakusumah, Asyifa, Auliya, & Ismail, 2024). World Federation for Mental Health (WFMH) says the mental health that is a condition which enable physical development, intellectual, and emotional, as long as it aligns with the social conditions and surrounding environment (Rozali et al., 2021).

Support Vector Machine (SVM) is a supervised machine learning algorithm used to address classification problems by determining the optimal hyperplane that maximizes the margin separating classes in an N-dimensional space. Based on solid mathematical principles from optimization theory, SVM has strong generalization performance and robustness to overfitting, especially in high-dimensional spaces. Its ability to handle both linear and non-linear problems effectively makes it widely applied across various fields. In this study, SVM is implemented to identify and analyze expressions in writing that reflect an individual's mental health.

FastAPI is a web framework based on the Python programming language, specifically designed for developing Application Programming Interfaces (APIs). It adopts Asynchronous Server Gateway Interface (ASGI) approach which enable to the invocation and execution endpoint asynchronously. FastAPI is the optimal choice for machine learning workloads, especially for developing Machine Learning platforms as a Service (MLaaS) (Suryotomo, Akbar, & Husaini, 2024).

The purpose of this research is to identify mental health conditions and indicate potential mental health disorders by analyzing text-based data from self-narrative or captions on social media. Sentiment analysis, or opinion mining, is the process of automatically comprehending, extracting, and processing data to obtain sentiment information from opinion-based sentences (Alexandridis, Varlamis, Korovesis, Caridakis, & Tsantilas, 2021). Sentiment analysis represents a significant area of study within the broader scope of Natural Language Processing (NLP) (Ardiani, Sujaini, & Tursina, 2020). NLP is used to construct the analysis framework, which is known as opinion mining or affective computing (Kanna & Pandiaraja, 2019; Rahanto & Kharisudin, 2021). The SVM classification model will be used to detect potential early mental health disorders, categorized into seven labels: normal, depression, suicidal ideation, anxiety, bipolar disorder, stress, and personality disorder.

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

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 5, **Number** 1, May 2025
https://doi.org/10.47709/brilliance.v5i1.6580

Based on the above discussion, this study uses SVM in creating a model capable of detecting mental health, disorders through sentiment analysis, implemented using the FastAPI framework. It is expected that this study will contribute significantly to supporting the early detection of mental health disorders.

LITERATURE REVIEW

Research on text-based mental health classification has been rapidly evolving, alongside the increasing volume of digital data reflecting individuals' psychological conditions, as captions on social media and digital interaction. One of the components for identifying mental health disorders through text analysis includes conditions such as depression, anxiety, and suicidal tendencies. Certain linguistic features, including the use of negative words, emotional phrases, and distinctive sentence patterns, have been proven to be useful in predicting an individual's mental condition (Pavlova & Berkers, 2022).

The SVM algorithm is a classical approach relevant to text classification, especially capable of handling large volumes of data and optimally separating classes. In study (Tadesse, Lin, Xu, & Yang, 2019), SVM demonstrates good and stable performance in detecting depression on social media platforms. This algorithm is superior in precision and computation time compared to other deep learning models, especially when dealing with limited data. In addition, integrating SVM with text representation methods like TF-IDF or word embeddings has been shown to enhance classification performance.

Regarding the implementation of the classification system, many studies have implemented Machine Learning as a Service approach (MLaaS) (Suryotomo et al., 2024). FastAPI is a Python framework designed to create API services effectively and efficiently. It is applied to machine learning models due to its ability to support asynchronous execution and its high compatibility with machine learning libraries such as Scikit-learn.

In this study, the Support Vector Machine (SVM) algorithm is employed as the primary method to perform text classification based on mental health conditions. SVM was chosen because it has the ability to handle large volumes of data and determine optimal values that separates classes effectively. Therefore, it is suitable for analyzing complex textual data with emotional characteristics. To ensure the model can be accessed and deployed practically, the resulting model will be integrated into an API framework. FastAPI is selected for this purpose because it offers an efficient, lightweight, and responsive solution for building Python-based APIs, and it is highly compatible with machine learning libraries such as Scikit-learn.

METHOD

SVM is one of the algorithms widely used in various machine learning studies for both classification and regression tasks. Initially, it was designed to address linear classification problems; however, as time progresses, it has evolved to handle non-linear problems by mapping data into higher-dimensional spaces, thereby enabling the identification of the optimal hyperplane that separates different classes (Abdusyukur, 2023). This algorithm has many capabilities, such as effectively handling datasets of both small and large scale, managing high-dimensional features, and being easy to implement (Gaye, Zhang, & Wulamu, 2021).

The study workflow is illustrated in Figure 1, detailing the processes of dataset acquisition, data preprocessing, and model development utilizing the Support Vector Machine (SVM) algorithm. Prior to system integration, the developed model undergoes rigorous evaluation to ensure its performance and reliability.

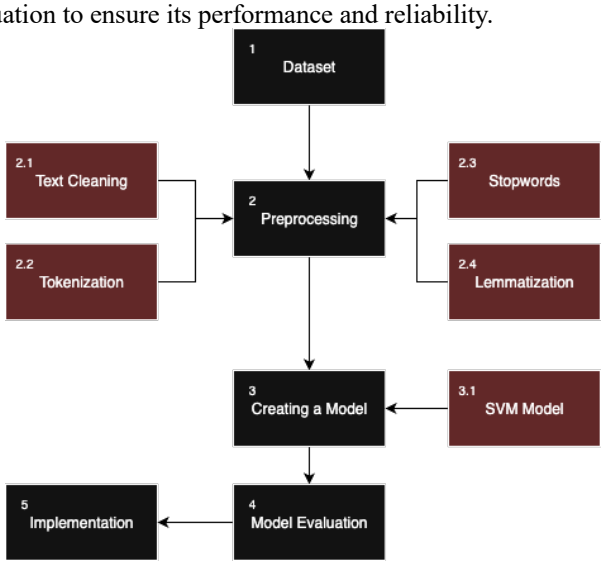

Fig 1. Stages of Research Methodology

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

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 5, **Number** 1, May 2025
https://doi.org/10.47709/brilliance.v5i1.6580

Dataset

The Mental Health are obtained from Kaggle (Sarkar, 2025), with a total of 53,043 data records. It includes the following attributes, an order number (no), the statement (text-comment sentence), and status (the respondent's mental health condition). The dataset is employed for sentiment analysis and model training utilizing the SVM algorithm to classify mental health conditions based on the textual inputs provided by respondents.

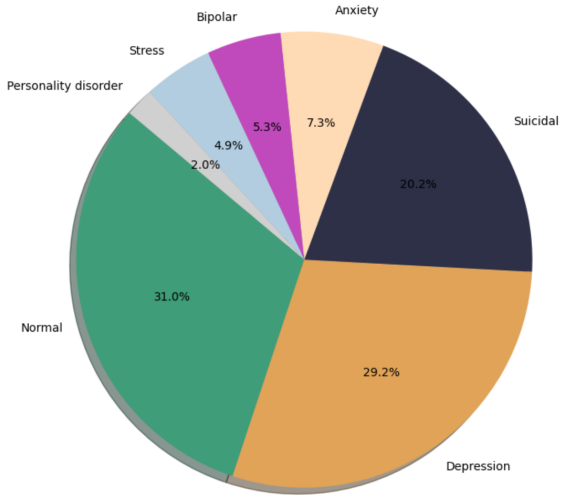

Fig 2. Distribution of Mental Health

Pre-processing data

The dataset obtained from Kaggle is in raw form and requires preprocessing, as it contains noise such as irregular punctuation and non-standard formatting. This step is essential to eliminate irrelevant information that could affect the model's performance and to prepare the data for training using the SVM algorithm. The following steps are applied in preprocessing.

1. Text Cleaning

   The first step in text cleaning is to remove irrelevant elements. This step aims to eliminate unnecessary components such as punctuation, mentions, hashtags, links, retweets, and numbers. Subsequently, case folding is applied to standardize the text format and ensure consistency (Darman, 2023). The preprocessing pipeline begins with converting all text to lowercase to eliminate discrepancies caused by case sensitivity, such as treating “Depression” and “depression” as different tokens. Subsequently, non-informative elements like URLs, mention tags (e.g., @username), and hashtags (#topic), which do not contribute semantic value to the content, are systematically removed. Further cleaning involves stripping punctuation marks, numeric characters, and other non-alphabetic symbols to reduce noise and enhance text quality. These steps collectively serve to normalize and purify the textual data, facilitating more accurate and efficient feature extraction in subsequent stages such as tokenization and vectorization. This rigorous cleaning process is critical for improving the robustness and generalizability of the machine learning model applied in later phases.
2. Tokenization

   After the cleaned data, the next step is tokenization. It is an initial process in text processing aimed at segmenting a sequence of characters in a sentence is divided into smaller units called tokens, usually words. (Albab, Fawaiq, & others, 2023). This process functions to identify word boundaries by considering the presence of specific delimiter characters. Common delimiters typically include whitespace characters such as spaces, tabs, and newline characters. However, certain punctuation marks—such as single quotation marks (`), periods (.), semicolons (;), and colons (:)—may also serve as word separators, depending on their contextual usage within the text.
3. Stopwords

   Tokens obtained from the previous step are filtered to remove stopwords. Stopwords are common words in English, such as “is,” “the,” and “and,” which frequently appear but do not carry specific meaning in text analysis. Removing stopwords helps reduce noise in the data and ensures that only meaningful information is retained for subsequent processing steps.
4. Lemmatization

   After stopwords are removed, each remaining token undergoes lemmatization. Lemmatization is a technique used to reduce words to their base or dictionary form—for example, “running” becomes “run” and

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

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 5, Number 1, May 2025
https://doi.org/10.47709/brilliance.v5i1.6580

"better" becomes "good." Unlike stemming, lemmatization preserves more natural and linguistically accurate word forms. This process helps unify morphological variations of words into a standardized form, enabling the model to recognize word meanings more efficiently.

Creating a Model

After the data has undergone preprocessing, the classification model is developed using the Support Vector Machine (SVM) algorithm. This algorithm is selected due to its advantages in handling high-dimensional data, such as textual data. During the training phase, the model is trained on cleaned and normalized data, enabling it to recognize linguistic patterns that represent each category of mental health conditions. The objective of this process is to produce a reliable predictive model capable of accurately classifying new, unseen text inputs.

Model Evaluation

Model performance is evaluated using metrics including precision, recall, and F1-score. Moreover, a confusion matrix was employed as a visual tool to illustrate the distribution of correct and incorrect predictions for each label, enabling the identification of misclassification patterns and highlighting the model's limitations in differentiating labels with similar features.

Implementation

In the implementation stage, the trained classification model was integrated into an application based on an Application Programming Interface (API) using FastAPI. FastAPI offers asynchronous API capabilities, allowing the program to run without waiting for each request and process to complete (Azhari, 2022). The model and label encoder were loaded to perform predictions on text input that had undergone the preprocessing stage, including normalization, removal of irrelevant characters, stopwords removal, and lemmatization. This system produces output in the form of mental condition status labels, a correspondence rate, and the probability distribution of each class, all delivered as a JSON response.

RESULT

Pre-processing data

Data pre-processing is performed to clean and prepare the text before it is used in the classification process. This process consists of four main stages, namely Text Cleaning, Tokenization, Stopwords Removal, and Lemmatization. The implementation of all these stages is shown in Fig. 3, which illustrates the preprocess_statement() function as the main process of text data cleaning.

```
def preprocess_statement(statement):
    statement = statement.lower()
    statement = re.sub(r"\d+|[^\w\s]|#\w+|@\w+|http\S+", "", statement)
    statement = statement.translate(str.maketrans("", "", string.punctuation))
    statement = statement.strip()

    # Tokenization
    tokens = word_tokenize(statement)

    # Stopword removal + Lemmatization
    cleaned_tokens = [
        lemmatizer.lemmatize(token)
        for token in tokens
        if token not in stop_words
    ]
    return " ".join(cleaned_tokens)
```

Fig 3. Text Cleaning, Tokenization, Stopwords Removal and Lemmatization

The code in Fig. 3 shows four processes, namely Text Cleaning, Tokenization, Stopwords Removal, and Lemmatization. The steps are as follows.

1. Text Cleaning

   At this stage, all text is converted to lowercase to maintain consistency, then links (URLs), punctuation marks, numbers, and special characters are removed using regular expressions. This process produces cleaner text that is free of elements irrelevant to the context of the analysis.
2. Tokenization

   After the cleaning process, the text is broken down into word fragments (tokens) using the tokenization method. This stage allows each word to be analyzed individually in the next stage.

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

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 5, **Number** 1, May 2025
https://doi.org/10.47709/brilliance.v5i1.6580

3. Stopwords Removal

   Frequent words that provide minimal contribution to the overall meaning, such as “and,” “or,” “that,” and other functional words, are removed from the text. The result of this process is that the text becomes more focused on meaningful words that are relevant to the classification context.

4. Lemmatization

   The final step is to convert each word to its base form. For example, the word “running” is converted to “run.” This process helps to unify variations of words that have the same meaning, thereby improving data consistency.

The end result of all these pre-processing steps is cleaner, standardized text data that is easier for classification models to process. Thus, the pre-processing stage improves the quality of the input used in model training and testing.

Creating a Model

After going through preprocessing and text data cleaning, the next step is building the classification model using the Support Vector Machine (SVM) algorithm. In its implementation, the data is divided with 80% used as training data and 20% as testing data to ensure the validity of the model evaluation. This ratio is intended to provide adequate data for training while preserving a representative portion for measure the model’s ability generalize to new, unseen data.

```python
# Split
X_train, X_test, y_train, y_test = train_test_split(
    X, y_encoded, test_size=0.2, random_state=42, stratify=y_encoded
)

# Pipeline with Calibrated SVM
base_svm = LinearSVC(max_iter=10000)
calibrated_svm = CalibratedClassifierCV(base_svm)

pipeline = Pipeline([
    ("tfidf", TfidfVectorizer()),
    ("svm", calibrated_svm)
])

# Train
pipeline.fit(X_train, y_train)

# Predict
y_pred = pipeline.predict(X_test)

# Accuracy
accuracy = accuracy_score(y_test, y_pred)

# Evaluation
print("\nClassification Report:")
print(f"Accuracy: {accuracy:.4f}")
print(classification_report(y_test, y_pred, target_names=label_encoder.classes_))
```

Fig 4. SVM Model

Model Evaluation

The evaluation of Support Vector Machines (SVM) shows an overall accuracy rate of 76%. This accuracy indicates a good level of classification capability in identifying various mental conditions based on text input. Further analysis of the classification report indicates that the 'normal' category label is the easiest for the model to recognize with an F1-score of 90%. This suggests that the model performs with a high level of effectiveness at distinguishing texts that do not suggest a mental disorder.

```
Classification Report:
Accuracy: 0.7645
                      precision    recall  f1-score   support

             Anxiety       0.83      0.77      0.80       768
             Bipolar       0.85      0.79      0.82       556
          Depression       0.70      0.73      0.71      3081
              Normal       0.86      0.94      0.90      3269
Personality disorder       0.81      0.63      0.71       215
              Stress       0.70      0.45      0.55       517
            Suicidal       0.66      0.63      0.65      2131

            accuracy                           0.76     10537
           macro avg       0.77      0.70      0.73     10537
        weighted avg       0.76      0.76      0.76     10537
```

Fig 5. Classification Report (SVM)

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

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 5, **Number** 1, May 2025
https://doi.org/10.47709/brilliance.v5i1.6580

The confusion matrix from the SVM model in the classification of seven mental condition labels shows good performance for the 'Normal' label, with 3,058 correct predictions. However, there are significant misclassifications in the 'Depression' and 'Suicidal' labels (578 and 629 errors), indicating similarities in data characteristics. This matrix can help identify the model's weaknesses in distinguishing between similar classes and serve as a basis for improving.

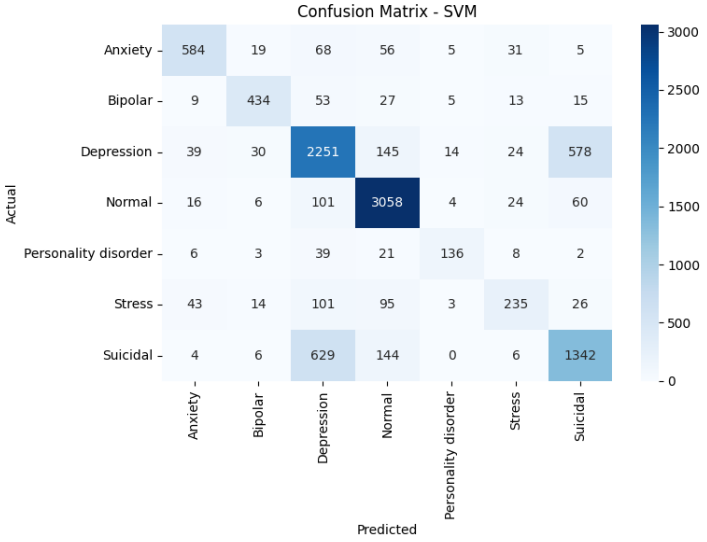

Fig 6. Confusion Matrix (SVM)

Confusion matrix in Fig. 6 from the Support Vector Machine (SVM) classification model. Based on these results, the model shows quite good ability in classifying several categories of mental disorders, especially in the Normal class with 3058 data successfully predicted correctly. Fairly good performance is also seen in the Anxiety class with 584 data classified correctly. However, there are still classification errors in several categories, such as 578 data in the Depression class that were predicted as Suicidal. The darker color pattern on the main diagonal indicates high accuracy in that class, while the lighter colors outside the diagonal indicate prediction errors between classes. Overall, this confusion matrix visually illustrates the distribution of the model's predictions and demonstrates the SVM's effectiveness in distinguishing between each category of mental disorder.

Implementation.

The trained classification model is integrated using the FastAPI framework, which builds a prediction system based on an API. Its integration is implemented by defining an endpoint named /predict, which accepts input in the form of a statement text from the user. This input is processed by the preprocess_statement function for cleaning before being passed to the model to generate a prediction. The result, consisting of the predicted class label and the corresponding probability score, is returned in JSON format, which can be used directly by user applications or other external systems.

```python
@app.post("/predict")
def predict(input_data: StatementInput):
    statement = input_data.statement.strip()

    if not statement:
        raise HTTPException(status_code=400, detail="Input tidak boleh kosong.")

    try:
        processed_statement = preprocess_statement(statement)

        # Prediksi kelas
        predicted_index = model.predict([processed_statement])[0]
        predicted_label = label_encoder.inverse_transform([predicted_index])[0]

        # Probabilitas semua kelas
        probas = model.predict_proba([processed_statement])[0]
        labels = label_encoder.classes_
        probas_dict = {
            label: f"{round(score * 100, 2)}%" for label, score in zip(labels, probas)
        }

        confidence_percent = round(max(probas) * 100, 2)

        return {
            "input": statement,
            "preprocessed": processed_statement,
            "predicted_status": predicted_label,
            "confidence": f"{confidence_percent}%",
            "class_probabilities": probas_dict,
            "model": "SVM"
        }
```

Fig 7. Prediction Endpoint

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

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 5, **Number** 1, May 2025
https://doi.org/10.47709/brilliance.v5i1.6580

Implementing a 'FastAPI' endpoint to generate predictions, the text input is first validated to ensure it is not empty. Then, the inference process is carried out using the trained model. In addition to the predicted label from the classification, the system also returns the probability score for each class. Figure 7 shows the test result using the statement 'I'm still worried' as input, which implies the 'Anxiety' label with a corresponding probability of 86.25%. This result demonstrates that the system can predict mental health disorders early based on text input.

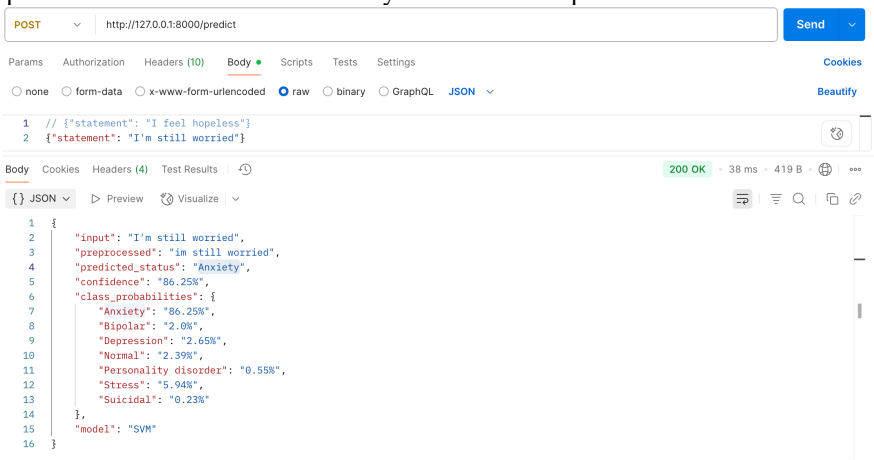

Fig 8. Test Results Using the FastAPI Framework

DISCUSSION

The outcome of the evaluation shows that the Support Vector Machine (SVM) algorithm achieved 76% accuracy in classifying text data associated with mental conditions. It demonstrates that SVM is capable of distinguishing between various mental healths categories with a good level of accuracy. Although not the most complex algorithm, SVM exhibits stable and reliable performance in the context of text classification.

Nonetheless, there are several limitations that need to be addressed, particularly in the interpretation of contextual and implicit expressions. For example, the statement “It felt like everything I did was in vain” reflects symptoms associated with depressive disorders, while “I often find it hard to breathe in certain situations, but I'm not sick” indicates symptoms of anxiety. However, the model misclassified these as belonging to the normal class. These misclassifications highlight the model’s inability to accurately detect emotional expressions that are conveyed implicitly through specific word choices.

This suggests that SVM still has limitations in handling contextual nuances in text. While SVM is known for its high accuracy in linearly separable and well-structured data, it tends to struggle with the ambiguity and subtlety of emotionally implicit language. Therefore, further analysis is required using data that includes implicit sentences, in order to improve classification accuracy.

CONCLUSION

The Support Vector Machine (SVM) is a machine learning algorithm used for mental health classification based on text. Its advantage lies in its ability to generalize to new data with high accuracy, making it a preferred option for sentiment classification in the context of mental health. The integration of this algorithm builds a classification model that can be used to analyze mental conditions and classify them into categories such as Normal, Depression, Suicidal, Anxiety, Bipolar, Stress, and Personality Disorder.

The implementation results of the SVM algorithm indicate that the model trained and tested on mental health condition data achieved an accuracy of 76%, demonstrating that the model performs reasonably well in classifying the data. It successfully processes textual data and classifies mental health statements based on the condition status. This research can be expanded by utilizing a larger and more diverse dataset, and by applying hybrid approaches or comparing with other machine learning algorithms to further improve the system's accuracy and reliability.

ACKNOWLEDGMENT

The author sincerely expresses gratitude for all the support received throughout the research process. Special thanks are extended to colleagues who contributed to the preparation and execution of this study, whether through scientific input, discussions, or technical assistance that facilitated the smooth progress of the research activities.

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

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 5, **Number** 1, May 2025
https://doi.org/10.47709/brilliance.v5i1.6580

REFERENCES

Abdusyukur, F. (2023). Penerapan Algoritma Support Vector Machine (Svm) Untuk Klasifikasi Pencemaran Nama Baik Di Media Sosial Twitter. *Komputa: Jurnal Ilmiah Komputer Dan Informatika*, *12*(1), 73–82.

Albab, M. U., Fawaiq, M. N., & others. (2023). Optimization of the Stemming Technique on Text preprocessing President 3 Periods Topic. *Jurnal Transformatika*, *20*(2), 1–12.

Alexandridis, G., Varlamis, I., Korovesis, K., Caridakis, G., & Tsantilas, P. (2021). A survey on sentiment analysis and opinion mining in greek social media. *Information*, *12*(8), 331.

Ardiani, L., Sujaini, H., & Tursina, T. (2020). Implementasi sentiment analysis tanggapan masyarakat terhadap pembangunan di Kota Pontianak. *JUSTIN (Jurnal Sistem Dan Teknologi Informasi)*, *8*(2), 183–190.

Azhari, R. Y. (2022). Web Service Framework: flask dan fastAPI. *Technology and Informatics Insight Journal*, *1*(1), 80–87.

Darman, R. (2023). Analisis Sentimen Respons Twitter terhadap Persyaratan Badan Penyelenggara Jaminan Sosial (BPJS) di Kantor Pertanahan. *Widya Bhumi*, *3*(2), 113–136.

Gaye, B., Zhang, D., & Wulamu, A. (2021). Improvement of support vector machine algorithm in big data background. *Mathematical Problems in Engineering*, *2021*(1), 5594899.

Kanna, P. R., & Pandiaraja, P. (2019). An efficient sentiment analysis approach for product review using turney algorithm. *Procedia Computer Science*, *165*, 356–362.

Pavlova, A., & Berkers, P. (2022). "Mental health" as defined by Twitter: Frames, emotions, stigma. *Health Communication*, *37*(5), 637–647.

Rahanto, F. F., & Kharisudin, I. (2021). Analisis sentimen data ulasan menggunakan metode naive bayes studi kasus the Wujil Resort & Conventions pada situs tripadvisor. *Unnes Journal of Mathematics*, 55–62.

Rozali, Y. A., Sitasari, N. W., Lenggogeni, A., Psikologi, F., Esa, U., Arjuna, J., … Kebon, T. (2021). Meningkatkan kesehatan mental di masa pandemic. *Jurnal Pengabdian Masyarakat AbdiMas*, *7*(2), 109–113.

Sarkar, S. (2025). Sentiment Analysis for Mental Health.

Supini, P., Gandakusumah, A. R. P., Asyifa, N., Auliya, Z. N., & Ismail, D. R. (2024). Faktor-faktor yang mempengaruhi kesehatan mental pada remaja. *Journal of Education Religion Humanities and Multidiciplinary*, *2*(1), 166–172.

Suryotomo, A. P., Akbar, B. M., & Husaini, R. (2024). Performance Analysis of FastAPI Framework on Lost Circulation Handling Management Application in Oil Well Drilling. *Telematika: Jurnal Informatika Dan Teknologi Informasi*, *21*(1), 110–121.

Tadesse, M. M., Lin, H., Xu, B., & Yang, L. (2019). Detection of depression-related posts in reddit social media forum. *Ieee Access*, *7*, 44883–44893.

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