Depression analysis of Twitter dataset using Natural Language Processing techniques

Siddamshetty Bhavesh Kumar
School of Computer Science and Engineering
VIT-AP University
Amaravati, Andhra Pradesh
bhavesh.siddamshetty@gmail.com

Jujjavarapu Sujan Chowdary
School of Computer Science and Engineering
VIT-AP University
Amaravati, Andhra Pradesh
chowdarysujan27@gmail.com

Karaka Rupasree
School of Computer Science and Engineering
VIT-AP University
Amaravati, Andhra Pradesh
rupasree200210@gmail.com

***Abstract*— Depression is a major global public health issue affecting many people. Twitter has become a popular platform for people to share their emotions and experiences, creating a useful source of data for understanding the sentiment and patterns related to depression. To analyze this data, we used natural language processing techniques and three machine learning models - Multinomial Naive Bayes, Support Vector Machines, and Logistic Regression. Our aim was to classify tweets into positive and negative sentiments, identify the most commonly used words and phrases, and explore how these patterns varied based on factors such as gender, age, and location.**

***Keywords*— *Logistic Regression, NLP, SVM, MultinomialNB, depression, Twitter, NLTK, TFIDF*.**

I. INTRODUCTION

Depression is a common mental health condition that affects millions of people worldwide and can lead to significant impacts on an individual's quality of life. In recent years, the use of social media platforms has increased, providing new opportunities to study mental health conditions such as depression through Natural Language Processing (NLP) techniques. This research paper focuses on using NLP techniques to analyze depression symptoms from Twitter data. The primary goal of the study is to develop a depression classification model that can accurately detect depression symptoms in Twitter data. To accomplish this, a dataset of tweets related to depression was collected, and three machine learning algorithms were used to build the classification model: Multinomial Naive Bayes, Logistic Regression, and Support Vector Machines.
The text data was transformed into numerical features using the TF-IDF vectorizer, and standard evaluation metrics such as accuracy, precision, recall, and F1 score were used to evaluate the model's performance.

The results indicated that all three models performed well in detecting depression in the dataset, with SVM achieving the highest overall performance. However, the models had a lower recall and precision for the depressed class, which suggests that additional data is required to improve their sensitivity and obtain good results.

The study emphasizes the potential of NLP techniques to analyze social media data for depression analysis. The insights gained from this research can be useful for mental health practitioners, researchers, and policymakers in developing effective strategies for early detection and treatment of depression. Additionally, the approach could be expanded to include other mental health conditions, and larger and more diverse datasets could be used to improve the generalizability of the models.
We discussed related work in depression analysis using NLP in Section II, the literature survey. The methodologies are discussed in Section III. The discussions that are based on the proposed model's results are discussed in Section IV. The model's future aspects and conclusions serve as the foundation for Section V.

II. RELATED WORK

Liza Wikarsa and Shrely Novianti Tharir [1] wrote a paper in 2015 which used Naïve Bayes Classifier on dataset which was acquired using Twitter API. NB Achieved 83% accuracy with a small number of tweets. It recommended using large data and other models for better accuracy. Sheeba Grover and Amandeeo Verma [2] wrote a paper in 2016 in which they used Rule based engine, SVM and NB on a Punjabi textual dataset. Their work is limited to Punjabi regional language text having a minimal scope. The integration of other machine learning models can achieve better results. Moin Nadeem [3] wrote a paper in 2016 in which he used DT, SVM, NB and LGR Classifier on lots of tweets. NB and LGR provide better results than the other techniques. Data ignores out-of-vocabulary words, and other techniques are required to identify depression from social media users' behavior. Maryam Mohammed Aldarwish & Hafiz Farooq Ahmad [4] wrote a paper in 2017 in which they used Rapidminer, SVM, NB on data acquired from Twitter and Facebook. Their work achieved good precision and the least accuracy. M Islam et al. [5] published a paper in 2018 in which he used DT, SVM, Ensemble and KNN on the data which was acquired from Face book. DT yields better accuracy in detecting depression. Work has limited scope. Michael M. Tadesse et al. [6] published the paper in 2019 in which he used NLP, LR, SVM, MLP,Random Forest, Adaptive Boosting on a dataset acquired from Reddit. SVM achieved 80% accuracy, 0.79 F1

scores with the bigram feature, and MLP attained 91% accuracy and 0.93 F1 scores with combined features. Their data is limited, and recommended to examine depression related behavior reflected in social media. Jina K et al [7] published a paper in 2020 in which they used CNN and XGBoost on a dataset obtained using reddit. Depression can be identified from posts on social media by combining deep learning techniques with the necessary natural language processing techniques. The work is limited to Reddit. Md Zia Uddin et al [8] published a paper in 2022 which used LSTM and RNN with LIME explainable AI on a dataset from Facebook, ung.no(pu). The work is limited to the data of young people.

III. Methodology

A. *Dataset*

For the dataset we have used two datasets from Kaggle. Kaggle is one of the best platforms for data scientists, machine learning engineers, and researchers to collaborate, share, and compete on data science and machine learning projects. It was founded in 2010 and acquired by Google in 2017. Kaggle provides a diverse range of datasets and challenges for users to explore, analyze, and develop models for various applications.
The two datasets we used are:
Depression: Twitter Dataset + Feature Extraction, a collection of tweets from Twitter. Each post is labeled with 1 for 'Depressed' and 0 for 'Not depressed'.
Sentimental analysis of Tweets, a collection of tweets from Twitter. Each post is labelled with 1 for 'Depressed' and 0 for 'Not depressed'.

We will call the first dataset as D1 and second as D2.
D1 consists of 20000 observations which has 10 attributes each such as 'post_id', 'post_created', 'post_text', 'user_id', 'followers', 'friends', 'favorites', 'statuses', 'retweets', 'label'.
D2 consists of 10314 observations and only 2 columns which are, message and label.

B. *Dataset Pre-processing and Exploration*

Then in the datasets we checked for any null values and then removed if any available. And then we sliced the D1 dataset to keep only tweet texts and labels and removed unnecessary columns as we will be working only with Tweets for this project as it is based on Natural Language Processing.
Then we checked the number of depressed and non-depressed posts for both the datasets and observed that the depressed count for the D2 dataset is relatively very low when compared to the non-depressed tweets. So, we will perform under sampling using RandomUnderSampler method from python. RandomUnderSampler is used to randomly remove samples from the majority class in order to balance the class distribution. This can be useful when the majority class contains a large number of samples, and the minority class contains only a few. RandomUnderSampler (sampling_strategy='not minority') will randomly remove samples from all classes except the minority class until the class distribution is balanced.

```
mental_health_tweets.isnull().all()
```

```
post_id         False
post_created    False
post_text       False
user_id         False
followers       False
friends         False
favourites      False
statuses        False
retweets        False
label           False
dtype: bool
```

```
sentiment_tweets.isnull().all()
```

```
message to examine           False
label (depression result)    False
dtype: bool
```

Figure 1 Null values in the given two datasets.

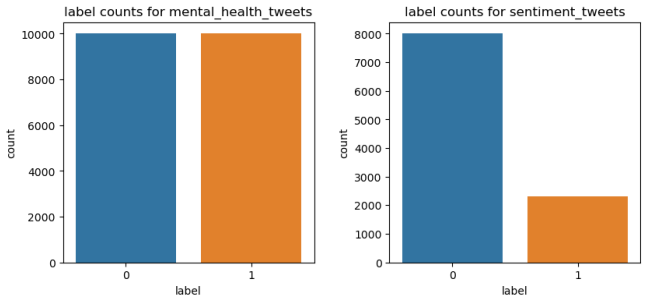

Figure 2 Count-plot for the number of tweets of each type.

After under-sampling the number of depressed and non-depressed tweets become 2314 each, and then we concatenate both the datasets. After concatenating both datasets, the new dataset becomes the size of (30314,2).
Then we calculate the word count in each row and add it as another column. We have also plotted the histogram of number of words per tweet.

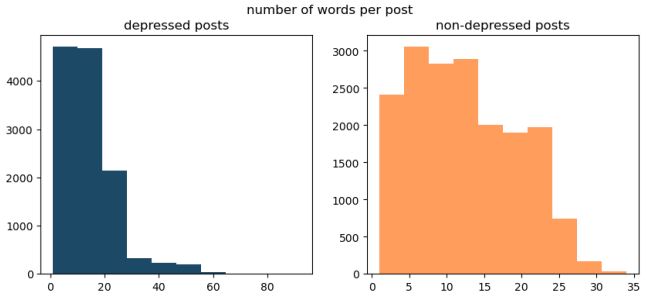

Figure 3 Number of words per post for each type of tweet.

From the obtained graphs we can deduce some important information of our datasets.

```
The minimum number of words in non-depressed posts is:  1
The maximum number of words in non-depressed posts is:  34
_________________________________________________________
The minimum number of words in depressed posts is:  1
The maximum number of words in non-depressed posts is:  92
_________________________________________________________
The average number of words in non-depressed posts is:  12
The average number of words in depressed posts is:  15
```

Figure 4 Data exploration.

Then before giving our data to the model, we need to split our data into training and testing sets. For that we have used scikit-learn library of python and divided the data into train-test splits of test size = 0.2 and random state = 10.

```
Number of training: 24251
Number of testing: 6063
```

Figure 5 Size of Training and Testing data.

Also, for the data exploration we have also made word cloud to understand what are the words which mostly used in depressed tweets and non-depressed tweets.

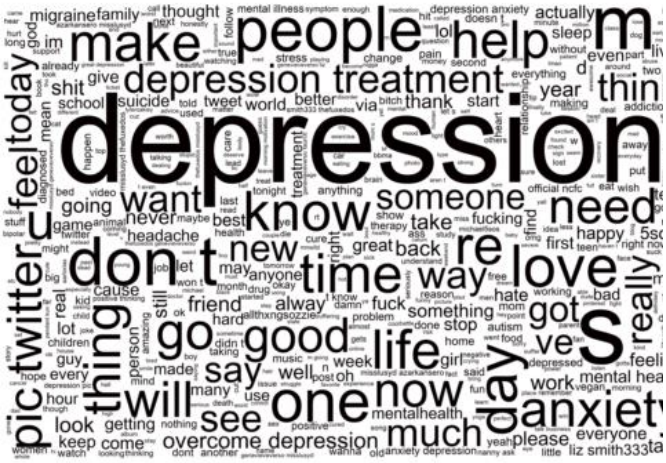

Figure 6 Word Cloud of Depressed Tweets.

Figure 7 Word Cloud for Non-Depressed Tweets

C. Text Cleaning and Pre-Processing

Text cleaning and pre-processing are important steps in natural language processing (NLP) that involve preparing text data for analysis. The goal of these steps is to remove noise, irrelevant information, and inconsistencies from the text, and to transform it into a standardized format that can be more easily analyzed and understood by machine learning models.

Text cleaning involves removing unwanted elements from the text such as stop words (common words like "the", "and", "is", etc.), punctuation, special characters, and digits. Removing these elements can help reduce the dimensionality of the text data and improve the performance of machine learning models.

Text pre-processing involves transforming the text into a standard format, which includes converting all letters to lowercase or uppercase, and removing stop words and accents or diacritics. Additionally, text pre-processing can involve techniques such as stemming and lemmatization, which are used to reduce words to their base form, text pre-processing includes removing HTML tags, URLs, and email addresses from the text, as well as correcting spelling mistakes and abbreviations. This can help reduce the number of unique words in the text data, making it easier to analyze and process.

For the text cleaning we have used the re module of python. The re module is a module that comes with Python, and it provides functionality for working with regular expressions, which are useful tools for matching patterns in strings. Using a specific set of syntax rules, developers can search, replace, and manipulate text as needed. This makes it possible to perform complex text processing tasks efficiently and accurately.

And for the text cleaning we have used the NLTK library from python. NLTK is a well-known open-source Python library for natural language processing that provides a variety of tools and resources for tasks such as tokenization, stemming, tagging, parsing, and sentiment analysis. A major advantage of NLTK is its versatility in processing many different natural languages, including English, French, Spanish, and German. The library also features useful resources like corpora and lexical resources, such as the Brown Corpus, WordNet, and the Penn Treebank, that are commonly employed in natural language processing research.

For text cleaning, we have defined a function which first converts the upper cases to lower cases and then removes the white spaces and then strips the HTML tags then removes all the http notations and retweet notations. After that we also included the removal of escape sequences and punctuations.

For the text pre-processing we have tokenized each sentence from each tweet in the dataset. And then we removed stop words from all the tokens received.

After removal of stop words we have lemmatized each and every token. Lemmatization is a crucial natural language processing technique that involves converting words to their base or root form (called the lemma). The primary goal of lemmatization is to standardize different forms of a word so that they can be treated as one item rather than as separate words. Lemmatization provides several benefits, including simplifying text data, which can make it easier to analyze and understand. Another advantage is that it can increase the accuracy of natural language processing tasks such as text classification, sentiment analysis, and machine translation. This is because lemmatization decreases the variations in word forms, which helps algorithms identify patterns and connections in text data more effectively.

Now we are done with the text pre-processing and we are ready to move to the next step.

D. Feature Extraction

In natural language processing, feature extraction involves converting raw text data into numerical features that can be understood by machine learning algorithms. The aim of feature extraction is to capture the relevant information in text

data and transform it into a format that machines can easily interpret.

There are several reasons why feature extraction is crucial in NLP. Firstly, it can help to reduce the dimensionality of text data, which can be extremely high due to the vast number of possible words and combinations of words. By selecting only, the most significant features, feature extraction can make it simpler and more efficient to analyze and model text data.

Secondly, feature extraction can enhance the accuracy of NLP models by focusing on the most informative aspects of the text data. By selecting features that are strongly associated with specific tasks, such as sentiment analysis or topic modeling, feature extraction can help to enhance the performance of machine learning algorithms on those tasks.

Lastly, feature extraction can facilitate the development of more resilient and adaptable NLP models. By selecting features that are relevant across multiple domains and languages, feature extraction can help to create models that can handle a wide range of text data instead of being limited to particular contexts or applications.

For this project we have decided to go with the Tfidf vectorizer for the feature extraction. TfidfVectorizer is a popular feature extraction technique used in natural language processing (NLP) for converting raw text into numerical representations that can be used for machine learning. TfidfVectorizer stands for Term Frequency-Inverse Document Frequency Vectorizer.

The TfidfVectorizer algorithm works by converting a collection of raw text documents into a matrix of TF-IDF features. TF-IDF stands for Term Frequency-Inverse Document Frequency and is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The TF-IDF value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus. This helps to adjust for the fact that some words are generally more common across all documents and therefore less informative for distinguishing between documents.

The TfidfVectorizer algorithm first tokenizes the raw text by breaking it up into words or phrases, known as tokens. It then applies a pre-processing step to the tokens, such as converting all words to lowercase, removing stop words, and stemming or lemmatizing the remaining words to reduce the number of unique words in the text. After pre-processing, the algorithm calculates the TF-IDF score for each word in each document. This results in a matrix of numerical values representing the importance of each word in each document.

The resulting matrix can then be used as input for machine learning algorithms, such as classification or clustering algorithms, to build predictive models based on the text data. TfidfVectorizer is commonly used for applications such as sentiment analysis, document classification, and information retrieval.

E. Classification Models

In our project, we have employed three types of classifiers, namely Multinomial Naive Bayes, Support Vector Machines (SVMs), and Logistic Regression. These classifiers are commonly used in machine learning and natural language processing (NLP) tasks.

Logistic Regression is a classification algorithm that models the probability of a binary outcome based on one or more predictor variables. It is often used in text classification tasks where the input variables are the frequency of words or other features in the document, and the output variable is the class label. Logistic regression is well-suited for handling both linearly and non-linearly separable data and is commonly used in NLP.

Multinomial Naive Bayes is a simple yet effective classification algorithm commonly used for text classification tasks. It works by calculating the probability of a document belonging to a particular class based on the frequency of words in the document. It is well-suited for handling discrete data such as word frequencies and is commonly used as a baseline model in NLP.

Support Vector Machines (SVMs) are a type of machine learning algorithm used for classification and regression analysis. SVMs find the best decision boundary to separate data points of different classes in a high-dimensional feature space, with the goal of maximizing the margin. SVMs can handle both linearly and non-linearly separable data and are well-suited for NLP tasks such as text classification, sentiment analysis, and named entity recognition. SVMs can also use kernel functions to map data into higher-dimensional spaces, allowing them to model non-linear relationships between features. This makes SVMs a popular choice for NLP tasks where the relationship between words and their meaning can be complex.

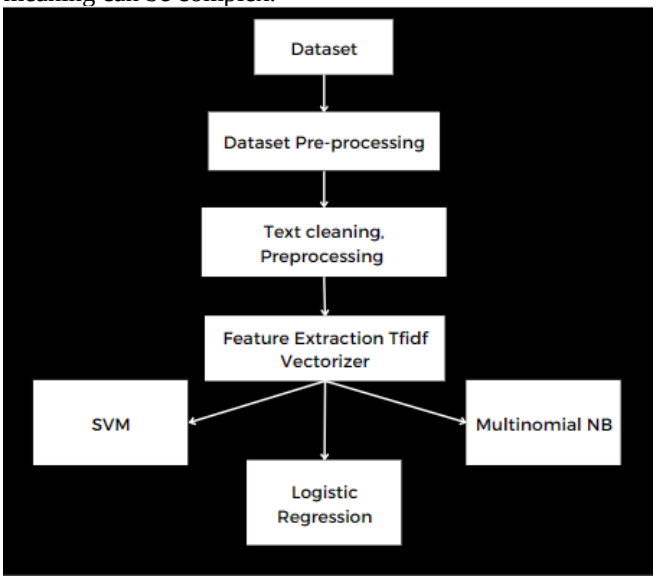

Figure 8 Model Architecture

IV. RESULTS AND DISCUSSIONS

This section mainly deals with the performance evaluation of the model. This model's performance metrics are recall, accuracy, f1-score, and precision. The above four methods are few best methods to evaluate a classification model. The percentage of cases when depressed and non-depressed are properly identified to the entire test set serves as a measure of accuracy. The ratio of cases that were classified as depressed

and non-depressed to all the instances that were classified in that manner is known as precision. Recall can be described as the proportion of cases that were classified as depressed or non-depressed to all instances those belonged to that category. The harmonic mean of recall and precision is the F1 score.
All the scores for each and every model are given by the Table 1.

S.no	Model Name	Accuracy	Precision	Recall	F1 Sco
1	Multinomial Naïve Bayes	0.81	0.87	0.65	0.7
2	Support Vector Machine	0.85	0.89	0.63	0.7
3	Logistic Regression	0.81	0.87	0.64	0.7

We have also used confusion matrices to evaluate our models. The confusion matrix is a critical tool for assessing the effectiveness of classification models. It is a tabular representation of a model's performance on a dataset, showing the number of accurate and inaccurate predictions by class. The matrix's rows indicate the actual data point labels, while the columns indicate the predicted labels.

The importance of the confusion matrix in machine learning lies in its ability to provide a detailed evaluation of a model's performance. The performance metrics derived from the matrix, such as precision, recall, accuracy, and F1 score, can help to understand how well the model is working, allowing for iterative improvements to be made.

The confusion matrix can also assist in detecting the types of errors that the model is making. Examining the matrix can indicate if the model is producing more false positives (Type I errors) or false negatives (Type II errors), making it easier to refine the model. In certain applications where the costs of false positives and false negatives are not equal, this analysis can be particularly beneficial.

Lastly, the confusion matrix can be used to compare different models' performance on the same dataset. By comparing the confusion matrices of multiple models, it is possible to determine which model performs best overall or which one is better at predicting certain classes.

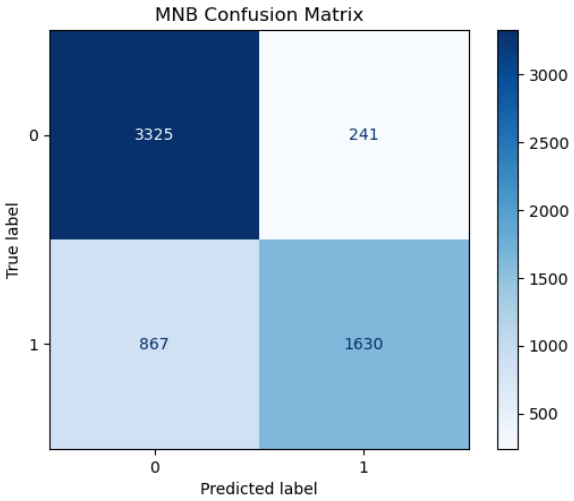

Figure 9 Confusion Matrix of Multinomial Naive Bayes

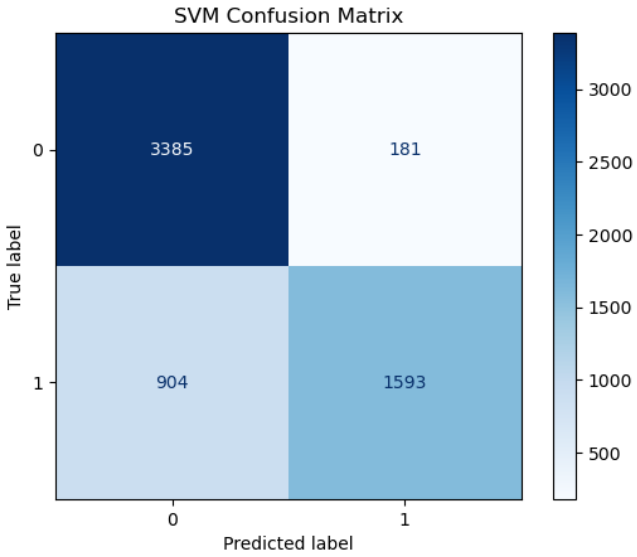

Figure 10 Confusion matrix of SVM

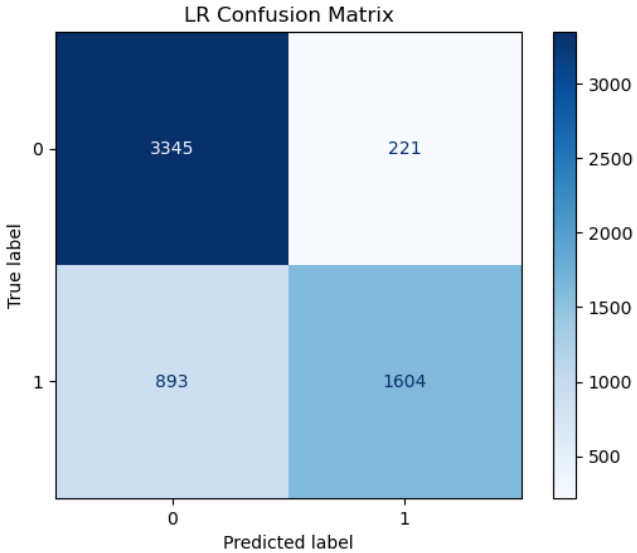

Figure 11 Confusion Matrix of Logistic Regression

From the above confusion matrices, we can observe that we are getting a small value of Type II error that is the number

of false negatives which are actually not good for our project as a Type II error (false negative) is generally considered worse than a Type I error (false positive). This is because a false negative could lead to a person with depression not receiving the appropriate care or treatment, potentially worsening their condition or putting them at risk. A false positive, while not ideal, would likely lead to further testing or assessment to confirm the diagnosis, rather than harm to the individual. So, we can say that our models are working properly and giving good results. And among the three models SVM is giving the best results.

V. Conclusion and Future Work

In conclusion, our Natural Language Processing (NLP) project has successfully built a depression classification model using Multinomial Naive Bayes, Logistic Regression, and Support Vector Machines, and two different datasets. To train and evaluate the models, we used the TF-IDF vectorizer to convert the text data into numerical features.

The performance evaluation of the models revealed that all three of them performed well in detecting depression in the dataset. The Support Vector Machines algorithm achieved the highest overall performance. However, the models had lower recall and precision values for the depressed class, indicating that more data is needed to improve their sensitivity and produce better results.

In the future, the project could be expanded by integrating other advanced techniques like word embeddings, neural networks, and ensemble methods to further enhance the classification performance. Also, by using larger and more diverse datasets, we can improve the models' generalizability. Furthermore, the project can be extended to classify other mental health conditions to provide more precise and comprehensive diagnoses of mental health issues using NLP.

References

[1] Wikarsa, L., & Thahir, S. N. (2015, November). A text mining application of emotion classifications of Twitter's users using Naive Bayes method. In 2015 1st International Conference on Wireless and Telematics (ICWT) (pp. 1-6). IEEE.

[2] Grover,S.,& Verma, A.(2016, August). Design for emotion detection of punjabi text using hybrid approach. In 2016 International Conference on Inventive Computation Technologies (ICICT) (Vol. 2, pp. 1-6). IEEE.

[3] Nadeem, M. (2016). Identifying depression on Twitter. arXiv preprint arXiv:1607.07384.

[4] Aldarwish, M. M., & Ahmad, H. F. (2017, March). Predicting depression levels using social media posts. In 2017 IEEE 13th international Symposium on Autonomous decentralized system (ISADS) (pp. 277-280). IEEE

[5] slam, M., Kabir, M. A., Ahmed, A., Kamal, A. R. M.,Wang, H., & Ulhaq, A. (2018). Depression detection from social network data using machine learning techniques. Health information science and systems, 6(1), 1-12.

[6] 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.

[7] Kim, J., Lee, J., Park, E., & Han, J. (2020). A deep learning model for detecting mental illness from user content on social media. Scientific reports, 10(1), 1-6.

[8] Uddin, M. Z., Dysthe, K. K., Følstad, A., & Brandtzaeg, P. B. (2022). Deep learning for prediction of depressive symptoms in a large textual dataset. Neural Computing and Applications, 34(1), 721-744.

[9] Amanat, A., Rizwan, M., Javed, A. R., Abdelhaq, M., Alsaqour, R., Pandya, S., & Uddin, M. (2022). Deep learning for depression detection from textual data. Electronics, 11(5), 676

[10] NLTK :: Natural Language Toolkit

[11] https://pypi.org/project/gensim/

[12] Iliev, A. I., Scordilis, M. S., Papa, J. P., & Falcão, A. X. (2010). Spoken emotion recognition through optimum-path forest classification using glottal features. Computer Speech & Language, 24(3), 445-460.

[13] Halfin,A. (2007). Depression: the benefits of early and appropriate treatment. American Journal of Managed Care, 13(4), S92.

[14] Zhang, T., Schoene, A. M., Ji, S., & Ananiadou, S. (2022). Natural language processing applied to mental illness detection: a narrative review. NPJ digital medicine, 5(1), 1-13.

[15] Leis, A., Ronzano, F., Mayer, M. A., Furlong, L. I., & Sanz, F. (2019). Detecting signs of depression in tweets in Spanish: behavioral and linguistic analysis. Journal of medical Internet research, 21(6), e14199.

[16] Jones, L.S., Anderson, E., Loades, M., Barnes, R., & Crawley, E. (2020). Can linguistic analysis be used to identify whether adolescents with a chronic illness are depressed? Clinical psychology & psychotherapy , 27(2), 179-192.

[17] Picardi, A., Lega, I., Tarsitani, L., Caredda, M., Matteucci, G., Zerella, M. P., & The, . D. (2016). A randomised controlled trial of the effectiveness of a program for early detection and treatment of depression in primary care. Journal of affective disorders, 198, 96-101.

[18] Rost, K., Smith, J. L., & Dickinson, M. (2004). The effect of improving primary care depression management on employee absenteeism and productivity a randomized trial. Medical care, 42(12), 1202.

[19] Dhand, A., Luke, D. A., Lang, C. E., & Lee, J. M. (2016). Social networks and neurological illness. Nature Reviews Neurology, 12(10), 605-612.

[20] Guntuku,S.C., Yaden, D.B.,Kern,M. L.,Ungar, L. H., & Eichstaedt, J. C. (2017). Detecting depression and mental illness on social media: an integrative review. Current Opinion in Behavioral Sciences, 18, 43-49.

[21] Kanwal, S., Malik, K., Shahzad, K., Aslam, F., & Nawaz, Z. (2019). Urdu named entity recognition: Corpus generation and deep learning applications. ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), 19(1), 1-13.

[22] Selva Birunda, S., & Kanniga Devi, R. (2021). A review on word embedding techniques for text classification. Innovative Data Communication Technologies and Application, 267-281.

[23] Tsugawa, S., Kikuchi, Y., Kishino, F., Nakajima, K., Itoh, Y., & Ohsaki, H. (2015, April). Recognizing depression from twitter activity. In Proceedings of the 33rd annual ACM conference on human factors in computing systems (pp. 3187-3196).