Journal of Data Analysis and Information Processing, 2023, 11, 420-453
https://www.scirp.org/journal/jdaip
ISSN Online: 2327-7203
ISSN Print: 2327-7211

A Stacking-Based Ensemble Approach with Embeddings from Language Models for Depression Detection from Social Media Text

Akwa Gaius[1], Ronald Waweru Mwangi[2], Antony Ngunyi[3]

[1]Department of Mathematics, Pan African University Institute for Basic Sciences, Technology and Innovation (PAUSTI), Nairobi, Kenya
[2]Department of Computing, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya
[3]Department of Statistics and Actuarial Sciences, Dedan Kimathi University of Technology, Nairobi, Kenya
Email: akwagaiusakwa@gmail.com, waweru_mwangi@icsit.jkuat.ac.ke, antonyngunyi@gmail.com

How to cite this paper: Gaius, A., Mwangi, R.W. and Ngunyi, A. (2023) A Stacking-Based Ensemble Approach with Embeddings from Language Models for Depression Detection from Social Media Text. *Journal of Data Analysis and Information Processing*, **11**, 420-453.
https://doi.org/10.4236/jdaip.2023.114022

Received: October 2, 2023
Accepted: November 7, 2023
Published: November 10, 2023

Abstract

Depression is a major public health problem around the world and contributes significantly to poor health and poverty. The rate of the number of people being affected is very high compared to the rate of medical treatment of the disease. Thus, the disease often remains untreated and suffering continues. Machine learning has been widely used in many studies in detecting depressive individuals from their contents on online social networks. From the related reviews, it is apparent that the application of stacking for diagnosing depression has been minimal. The study implements stacking based on Extra Tree, Extreme Gradient Boosting, Light Gradient Boosting and Multi-layer perceptron and compares its performance to state of the art bagging and boosting ensemble learners. To better evaluate the effectiveness of the proposed stacking approach, three pretrain word embeddings techniques including: Word2vec, Global Vectors and Embeddings from language models were employed with two datasets. Also, a corrected resampled paired t-test was applied to test the significance of the stacked accuracy against the baseline accuracy. The experimental results shows that the stacking approach yields favourable results with a best accuracy of 99.54%.

Keywords

Machine Learning, Natural Language Processing, Depression

1. Introduction

1.1. Depression

Health is not just the absence of disease, but a state of complete mental, physical

and social well-being [1]. On the other hand, mental health is a state of well-being in which the individual can cope with the normal stresses of life, realizes his or her own abilities and can work fruitfully [1]. Mental illnesses, such as depression and Post-traumatic stress disorder (PTSD) cause a large share of the burden of disease worldwide [2] [3], but are under diagnosed and undertreated around the world [4]. Depression is a major form of mental health disorder that negatively affects the way you think, how you feel and how you act. Some of its signs and symptoms include: feeling sad, having a depressed mood, changes in appetite, loss of pleasure or interest in activities once enjoyed, sleeping too much or difficulty in sleeping, weight loss or gain unrelated to dieting, loss of energy or increased fatigue. It affects all age groups, both men and women [5].

Depression is an important risk factors for other lifestyle diseases like ischemic heart diseases, hypertension, diabetes, as well as intentional and unintentional injuries [5] [6]. It can lead to suicide and other communicable diseases like Tuberclosis [6]. Individuals suffering from the illness often perform poorly in work places and educational institution. As a result, are deprived of many social and economic privileges and opportunities [5]. The economic burdens of depression are enormous and often unmeasured. Depression contributes significantly to poor health and poverty.

Depression has become the fourth major disease globally [7]. Compared to the number of people who are being affected, the rate of medical treatment of depression is very low due to difficulty of diagnosing the illness [7]. According to the World Health Organization survey in 2012, more than 350 million people were suffering from depression and almost 1 million people with depression commit suicide each year [7]. The lifetime prevalence of depression among high and low-middle income countries is estimated to be 14.6% and 11.1% respectively. More than 20% of the population of the united state experience a mental health illness in a given year [8]. About 11% of the population of Europe died of having depression related illnesses and 9.3% of the Chinese population suffer from depression related illnesses [8]. An organisation for economic co-operation and Development report in Japan shows that the standardized suicide rate per hundred thousand people in Japan was 20.9% in 2011 [9]. Mental health problems are critically increasing in Africa too. From 2000 to 2015, the lost years of disability increased by 52% due to mental and substance use disorder while the continent's population grew by 49% [8].

1.2. Background of the Study

1.2.1. Natural Language Processing, Machine Learning and Ensemble Learning

Natural language processing (NLP) and machine learning has been utilized enormously in recent years in depicting the mental state of users based on their shared content in online social media networks. NLP is a branch of artificial intelligence that gives machines the ability of read, understand and derive meaning from human language. It combines computer science, linguistics and machine

learning to create models that can comprehend, break down and separate significant details from text and speech. On the other hand, Machine learning (ML) involves learning patterns from data and making predictions based on those patterns. Supervised learning is one subcategory of Machine learning which involves the use of labeled datasets to train algorithms to classify data accurately. Regression and classification are two categories of supervised learning.

Ensemble learning involves employing a set of base (weak) learners, training them on a given training data, and combining their predictions to get one final prediction [10]. A weak learner is a learner who performs slightly better than a random guess [11]. The base learners can be trained on different training sets or feature sets and the combination of their predictions aims to reduce overfitting problems, and improve the accuracy and generalization ability of classification [12]. The base predictions are combined by voting or meta-learning [10]. In meta-learning, there is more than one learning stage and it involves learners learning from the output of other learners to output the final prediction. Max voting, averaging voting and weighted average voting are the three methods of voting. Max voting involves outputting the class with the most votes. Averaging voting involves averaging the base predictions by determining their arithmetic mean while weighted average voting involves averaging by assigning different weights to base predictions. The most popular ensemble learning techniques are bagging, boosting and stacking [10].

Bagging [13] is an ensemble learning technique that combines several base learners trained on multiple small subsets of the original data. To develop a bagging model, multiple datasets are generated by bootstrapping the original dataset and individual algorithms are trained on the bootstrap replicates. Predictions are made with the developed models and the final prediction is gotten by majority voting of the various predictions. Random Forests (RF) algorithm is an example of bagging. Boosting [14], on the other hand, is a sequential process that attempts to convert several weak learners to a strong learner. Each weak learner tries to correct the weaknesses of the previous model by assigning more weights to samples that were wrongly identified by the predecessor. Examples of boosting algorithms are Adaptive Boosting (AdaBoost) and Extreme Gradient Boosting (XGBoost).

The stacking method was introduced by Wolpert [15], and is also referred to as stacked generalization. The stacked architecture consists of two or more base learners, also known as the level 0 learner and a meta learner, also referred to as the level 1 learner. The meta learner learns how to best combine the predictions of the base models, trained on a given training data. For classification, these outputs from the base models can be class labels or probability values [16]. Stacking performs as well as or better than the individual methods that constitute the system [17]. Like other ensemble methods, stacking is also computationally costly.

1.2.2. Social Media Network

The online use of social networks such as twitter, Instagram, Facebook, Whatapp

have been in a rise in recent years. The site has over 200 million users active each month [18], generating messages at a peak rate of 230,000 per minute [18]. These platform gives users the opportunity to express their opinion, feelings, emotions, ideas and sentiments about a given topic, situation or issue online. While this is useful for users on one hand, it also provides an opportunity or platform for experts to monitor the mental state of individuals based on the written expressions and activities online. Machine learning has widely been used in depicting the mental state of users based on their content on social networks with significant positive performance.

This work focuses on ensemble and supervised machine learning strategy based on stacking to identify individuals with depression from their text from online social media networks. The related review shows that there has been little progress in using stacking techniques to identify depression in social media networks; hence, more study and analysis are needed. Also, a number of current studies for depression detection from text used ensemble models or neural networks. However, there hasn't been enough focus on combining ensembles and neural networks in a single framework. The following are the study's contributions:

- Application of Stacking based on three ensemble learning approaches and a deep learning approach namely; Extremely Randomized Trees (ET), XGBoost, Light gradient boosting machine (LightGBM) and Multi-layer perceptron (MLP). To the best of the authors' knowlegde, these combinations have never been used in any other study. Despite the ET, XGBoost and LightGBM models' potential for learning and their strong mathematical theory, they can only employ tree models that belong to the same category, and it is challenging to overcome the tree models' shortcomings. The MLP is used to enhance the training models' performance while aiming to generate better results.
- Application of feature extraction techniques including: Word2vec, Global Vectors (GloVe), and Embeddings from language models (ELMo) to establish real-valued vector representations of documents on two datasets namely; Sentiment_tweets3 and depression_data_reddit_clean. The latter dataset has not been identified in any other paper to the best of the authors' knowledge.
- Application of performance evaluation measures including: accuracy, precision, recall, F1-measure and Area under the receiver operating characteristic curve (AUC) to assess the performance of all classifiers.
- Comparison of the feature extraction approaches.
- Comparison of the suggested stacked ensemble technique to MLP and other bagging and boosting strategies including: RF, ET, Adaboost, Gradient Boosting Machine (GBM), XGBoost, and LightGBM, as well as other works in literature.
- Application of a corrected resampled paired t-test to assess the accuracy significance of the proposed stacking strategy. This method has never been utilized in any other study for depression detection from text.
- The remainder of the study is structured as follows. The prior research and

methods for identifying depression from social media text are reviewed in Section 2. Shared in Section 3 are the methodologies used in this investigation. Section 4 explains the findings, and Section 5 offers the study's conclusions.

2. Related Works

Several feature extraction and data mining techniques have been employed by researchers in diagnosing depression from scrapped data from different social media platforms across different languages. Some restricted to text data only while some extend to other user information in these blogs.

[19] compared the performances of several K-Nearest Neighbors (KNN) classification algorithms including: Fine KNN, Coarse KNN, Medium KNN, Cubic KNN, Cosine KNN, and Weighted KNN. Four feature sets were created based on Linguistic Inquiry and Word Count (LIWC) processes including: the linguistic style, temporal process, emotional process, and all three categories. The Coarse KNN outformed the other KNNs based on precision, recall and F1-score on the different feature sets. [20] employed several techniques including: Decision tree (DT), Support Vector Machine (SVM), KNN, Boosting and Bagging to detect depression among social media users. Feature extraction was based on LIWC. Four feature sets were created based on linguistic styles, temporal processes, emotional processes and a combination of all three. The DT performed better than the other classifiers employed with an accuracy of 72%. [21] used SVM and RF with different feature extraction techniques including Term frequency-inverse document frequency (TF-IDF), word embeddings and Bag of Words (BoW) as well as morphological and stylometric features for detecting depression from social media data. The SVM model with TF-IDF plus morphological and stylometric features yielded the best F1-score of 63.36%, while TF-IDF with RF yielded the best precision of 79.4%.

[22] implemented RF for diagnosing depression from text data based on LIWC processes as well as temporal and non-temporal emotion features. The RF achieved a best accuracy of 87.27% with non-temporal feature vectors which consist of 9 entries for each user (Emotion Overall Score, Disgust, Anger, Happiness, Fear, Surprise, Sadness, Confusion, Shame). [23] implemented the regularized generalized linear model with TF-IDF for detecting Nigerians with depression from their tweets, with a precision, recall and F1-score of 0.89, 0.91 and 0.90 respectively. [24] developed a majority vote ensemble for depression detection from twitter data based on three baseline classifiers including: Naive Bayes (NB), GBM and RF. Feature extraction was based on 5 feature sets related to posting patterns and linguistic cues and the majority vote ensemble classifier outperformed the baseline classifiers with accuracy of 85.09%.

[25] used the hierarchical hidden Markov model for calculating the degree of depression of users in online social communities. Next, Logistic Regression (LR), RF, and Gaussian NB were implemented with different word representation techniques including: GloVe, Word2Vec and FastText. RF with FastText achieved

the best performance. [26] applied LR, SVM, AdaBoost, RF, and MLP on Depression-related text post obtained from Reddit. Feature extraction was based on N-gram, Linear discriminant analysis (LDA) and LIWC. [27] applied sentiment analysis on tweets to assigned binary classes. That is, the depressed and non-depressed class. The NB and SVM algorithms were employed with BoW for feature extraction and the result showed that the NB performed better than SVM.

[28] used TF-IDF to developed a linear kernel SVM classifier based on users tweets and their activities on twitter. The classifier scored an accuracy and recall of 82.5% and 85% respectively.

[29] used several classifiers including RF, LR, SVM, and NB for detecting depression from social media content. The results obtained shows that the LR provides higher performance. [30] implemented convolution neural network (CNN) classifier with bidirectional encoder representations from transformers (BERT) for obtaining embeddings and achieved an accuracy, precision, recall and F1-score of 88.4%, 90.3%, 87%, and 93.6% respectively.

[31] implemented XGBoost and CNN for detecting mental illness using TF-IDF and Continuous bag of words (CBOW) feature extraction techniques respectively. [32] implemented long short-term memory (LSTM) on tweets with 93% accuracy. Vectorization of the dataset was done by generating features based on the maximum length of the tweets and an online dictionary for mapping words to numerical values. [33] applied RF, XGBoost, NB, SVM, KNN and DT algorithms with BoW feature extraction technique for predicting depression from Bengali text. The NB achieved the best accuracy of 86.6%.

[34] employed several classifiers including NB, LR, SVM and LSTM with word2vec embedding technique for identifying depression. The LSTM outperformed the other models with an accuracy of 99.92%. [35] suggested a hybrid method to evaluate user text messages on Reddit for detecting depression. Deep learning techniques were trained on the created training data, and their effectiveness was evaluated using the test data. In particular, the Bidirectional LSTM (BiLSTM) was proposed, which included a number of metadata elements and word embedding techniques, and it achieved favorable performance. [36] used the python textblob package to assigned sentiments labels to twitter depression related post and python NLTK package to extract 86 linguistic features such as sentence-level, word-level and character-level features. Several traditional and ensemble learning techniques including NB, DT, RF, SVM, LR, Adaboost, Bagging, Stacking and Multilayer Perceptron were applied and the RF outperformed the other classifiers with an accuracy of 60.54%

[37] implemented majority voting and blending for diagnising depression from tweets based on LR, KNN, DT, SVM, NB and MLP and achieved an accuracy of 85.35% and 87.21% respectively. [38] used NB, KNN, SVM, regularized LR and Simple LR to implement stacking for detecting depression from Chinesse social media data based on attributes related to one of seven categories (part of speech, emotional words, personal pronoun, specific words, polarity, posting

habits, posting time) with a best accuracy of 90.27%. [39] suggested a hybrid model based on CNN and biLSTM with an accuracy of 94.28% for predicting depression from tweets. The convolutional layer of CNN was used to extract embeddings while the biLSTM was used to analyze longer text sequences. Comparisons were made between the proposed CNN-BiLSTM model, CNN and RNN models, as well as other standard techniques. Experimental results using a number of performance metrics suggest that the proposed technique can improve predictive performance.

[40] applied bagging on three datasets based on LSTM and LR for identifying individuals suffering from depression. Averaging was used to fuse the results of the classifiers. Feature extraction was based on Glove embeddings and sentiment features for the LSTM and LR respectively and the ensemble classifier achieved a best accuracy and precision of 75.55% and 85.05% respectively. [41] implemented Adaboost, DT, KNN, MLP, LR, NB, SVM, linear discriminant analysis (LDA) and RF for diagnosing depression from social media text on a manually annotated dataset of three classes. Feature extraction was based on TF-IDF, Word2vec and GloVe. The RF with Word2vec embeddings outperformed the other classifier with an accuracy of 87.7%. [42] applied KNN, NB, LR, SVM, RF, CNN and Gated Recurrent Unit (GRU) for detecting severity of depression from Bengali text. The data was extracted from Bengali social media platform and annotation was based on the "Diagnostic and Statistical Manual of Mental Disorders". Feature extraction was based on TF-IDF for the traditional Machine Learning techniques and word embeddings for the deep learning techniques. The GRU achieved the best accuracy of 81%.

[43] made a contrast between Stemming, Lemmatization and different lexical techniques in a suggested arabic twitter preparation system. The twitter data used was retrieved from twitter and annotation was done by five experts. Next, Universal Sentence Encoder (USE) and BERT were applied. Different performance measures including accuracy, F1-score, specificity, Youden Index, AUC, Intersection over Union (IoU), and weighted sum metrics (WSM) were used to access the models' performance. The Arabic BERT models achieved the best WSM of 95.26%, while the USE models achieved the best WSN of 80.20%. [44] proposed a sentiment analysis method based on man-made criteria and terminology for calculating the inclination of depression. Next, a predictive framework was developed which incorporates a CNN model built on a benchmark twitter dataset and Whale Optimization Algorithm for hyperparameters optimization during training. The proposed approached outperformed BiLSTM and CNN-BiLSTM with a recall of 92.89%. [45] implemented several Machine learning classifiers including the NB, LR and SVM. Feature extraction was based on TF-IDF, GLove and all-MiniLM-L6-v2. For each feature extraction technique, additional sentiment features were added which include the subjectivity, polarity, positivity, neutrality and negativity of the text with the aid of the Vader and textblob package of scikit-learn library of python. The experimental results show that the classifiers achieved a better results with the addition of the sentiment

features.

[46] proposed a feature for detecting depression from tweets based on the average of the sentiment value of the word that make up the tweets and showed that this feature is useful to diagnose depression by training an XGBoost classifier with accuracy of 78%. When combined with N-Gram + TF-IDF and LDA, an accuracy of 89% was achieved using the SVM classifier. [47] proposed the SERCNN framework for identifying individuals with depression from their tweets. The proposed framework was built by stacking two pretrain Glove embeddings trained on different domains and reintroducing the concatenated output to the CNN classsifier. The proposed framework outperformed other baselines and state-of-the-art approaches with an accuracy of 93.7%. [48] applied six machine learning algorithms for diagnosing depression from tweets. Feature extraction was based on TF-IDF and the SVM outperformed all other classifiers with an accuracy of 79.90%.

[49] first applied LSTM for detecting depression from tweets. Feature extraction was based on fast text word embeddings and the classifier achieved a 95.12% accuracy. Next, the models' performance was enhanced using a hybrid BiLSTM + CNN model. Superior sequence features were extracted from text data using the BiLSTM layers whereas CNN improves performance with higher dimensional features. [50] proposed a novel classification model called the cost-sensitive boosting pruning tree (CBPT) based on Adaboost on two depression datasets. The pruning item helps to determine the optimal depths and leaves of the tree model while the cost item helps to assign different weights to misclassified samples based on the level of difficulty in classifying them in the previous iteration. Feature extraction was based on LDA, user profile feature, social interaction feature and linguistic features. In comparison to numerous state-of-the-art boosting algorithms, CBPT achieves pleasing classification results

[51] proposed a hybrid deep learning classifier of accuracy of 0.86 and an F1-score of 0.86 which constitutes a pretrained sentence BERT and CNN to diagnose individuals suffering from depression using the Self-reported Mental Health Diagnoses dataset. BERT was used to obtain the embeddings of meaningful information in every post while CNN was used for more transformation of these representations and the temporal establishment of behavioral patterns of users. [52] applied LSTM and several machine classifiers including Multinomial NB, RF and SVM for depression detection task from twitter data. The experimental results showed that the SVM achieved the best accuracy of 80.37% while the LSTM achieved the best recall of 0.88%. Feature extraction was based on TF-IDF for the SVM classifiers and Word2vec for LSTM. [53] compared the performances of SVM, KNN, RF, DT, LR and NB on detecting individuals with depression from Facebook and YouTube comments. The retrieved posts were manually annotated as either being depressed or not depressed and feature extraction was based on TF-IDF. The SVM outperformed the other classifiers with an accuracy of 75.15%. [54] trained an emotion detection model from speech based on LSTM with 98% accuracy and a depression detection model from

tweets based on RF with 95% accuracy. Feature extraction was based on the Mel Frequency Cepstral Coefficients (MFCC) for the emotion model and TF-IDF for the depression model. The prediction obtained from both classifier were combined with a DT which resulted in a 100% accuracy. [55] applied bagging based on DT with BoW for depression detection from Twitter data. The experimental results showed that the proposed approach outperformed other classical machine learning techniques with an accuracy of 98.33%. [56] implemented LR, SVM, RF and gradient boosting for detection of students aged 10 to 17 with depression based on their written compositions and scores on a Children's Depression Inventory. Feature extraction was based on all 102 processes of the Chinese version of LIWC and a pre-train corpus consisting of 300-dimensional Word2vec vectors.

3. Methods

3.1. Datasets

Data can be accessed freely from social media networks but much time and expertise are required to clean and annotate texts for depression. Two datasets that are freely and publicly accessible have been used in this study: Sentiment_tweets 3 and Depression_dataset_reddit_cleaned dataset.

Sentiment_tweets 3: It is a benchmark dataset for depression prediction containing 10,314 labeled instances. It consists of two main features: "message" and "label". The "message" feature contains tweets in english and not restricted to any demographic characteristics of the users. While the "label" feature indicates whether a tweet is classified as depression or not. A label of "1" for a tweet denotes that the user who posted the tweet has depression while a label of "0" denotes otherwise. 8000 of the instances are non-depressive (labeled 0) and are randomly sampled instances with positive sentiment from the Sentiment_140 dataset of [57], obtained using the Twitter application programming interface (API) with the help of related keywords. The remaining 2314 instances are depressive (labeled 1) and were scrapped from Twitter using the Twitter Intelligence Tool (TWINT) with the aid of keywords and hashtags related to depression. The dataset is avalaible at [1]. The Sentiment_140 is one of the most well-known datasets for sentiment analysis. It consists of 1.6 million tweets, and has annotation as either being 4 = positive, 2 = neutral or 0 = negative.

Depression_dataset_reddit_cleaned: It consist of a subset of randomly selected instances from the Reddit Self-reported Depression Diagnosis dataset developed in [58] due to its large size and has been cleaned using a variety of NLP techniques. It consists of two main features: "clean_text" and "is_depression". The "clean_text" feature contains posts of Reddit users not restricted to any demographic characteristics. While the "is_depression" feature indicates whether a post is labeled as depression or not. A label of "1" for a post denotes depression while a label of "0" denotes otherwise. Annotation of the dataset was done by

[1] https://raw.githubusercontent.com/viritaromero/Detecting-Depression-in-Tweets/master/sentiment_tweets3.csv.

three experts. It consists of 7731 instances, and 3831 of which are labeled as depressive and 3900 as non-depressive. The dataset only captures a subpopulation of Reddit users and may not be a representation of the whole population. It is available at [2].

3.2. Data Preprocessing

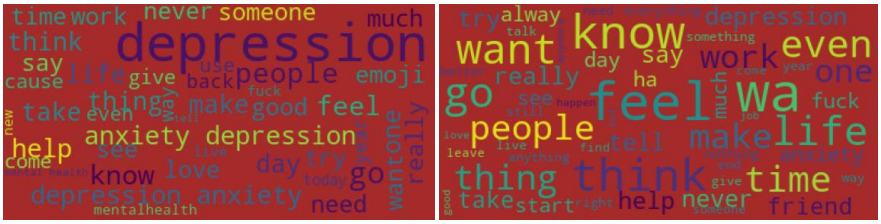
Figure 1 shows the most common words for the datasets used by depressed users after data preprocessing. Data preprocessing helps to produce a cleaner and more manageable dataset for subsequent analysis. Thus, enabling a more accurate and efficient transformation of documents into vectors and enhancing the performance of the learning models. The steps include:

- Removal of stop words. Stop words are regularly occurring words such as "a", "the", "is". They do not provide useful information.
- Removal of noise from the text to improve purity by eliminating unreadable characters such as URLs and non-ASCII characters.
- Words in social media maybe spelled incorrectly or abbreviated. We employed the Textblob API described in [59] to correct misspelled words.
- Emoji and emoticons in texts also provide valuable information and are converted to their respective words for inclusion in the analysis.
- Conversion of all characters to lowercase to avoid case sensitivity.
- Lemmatization, which involves the conversion of words to their base form to reduce ambiguity.
- Tokenization, which involves the spliting of text to individual words.

3.3. Feature Extraction

Pre-train Word2vec, GLoVe and ELMo word embeddings have been used for feature extraction in this work. Word embeddings are real-valued vector representation of words such that words with similar meaning have similar representation. That is, are closer in the vector space. Word2vec and GloVe are traditional embeddings techniques and thus, assigns a single vector representation to each word. Whereas ELMo is a contextualized embedding and thus, generates dynamic enbeddings of words based on the context in which they appear. Thus, by employing these techniques, we attempt to compare the effectiveness of these two categories of embeddings on depression detection task. Also, these approaches have shown to yield favourable results in many text classification tasks. The fact that pre-trained embeddings are learned on a large corpus and can yield significantly better performance when implemented on smaller training datasets is the primary reason of utilizing them in this work. So, using pre-trained embeddings, we attempted to compare the effectiveness of the stacking technique to the baseline ensemble learning techniques. In this study, document (text) embeddings were obtained by averaging the embeddings of the words that make up the document.

[2] https://www.kaggle.com/datasets/infamouscoder/depression-reddit-cleaned.



(a) Sentiment_tweets3 (b) Depression_dataset_reddit_cleaned

Figure 1. Word cloud of depressive post.

3.3.1. Word2vec

Word2vec [60] is a neural network model that produces embeddings of words based on the assumption that the semantics of a word can be inferred from its context. In essence, it seeks to maximize the probability of the target term given the input term as

$$p\left(w_j / w_k\right) = \frac{\exp\left(v'^{\mathrm{T}}_{w_j} v_{w_k}\right)}{\sum_i^v \exp\left(v'^{\mathrm{T}}_{w_i} v_{w_k}\right)} \tag{3.1}$$

where w_j and v'_{w_j} are the target word and its word embedding respectively. w_k and v_{w_k} represents the input word and its word embeddings respectively. v is the vocabulary size. T and exp denotes the transpose and exponential respectively.

Word2vec embeddings can be estimated by two approaches namely; The Continuous Bag-of-Words(CBOW) approach, which learns the embedding by estimating the target word based on its context and the Skip-gram approach, which learns by estimating the context based on an input word. Figure 2 is an illustration of the Word2vec techniques.

In this study, we used the Google news corpus which consists of a pre-trained vector representation of words obtained by learning the CBOW model on more than 100 billion words from the Google News dataset. It consists of 300-dimensional vectors of 3 million words and phrases and is available at [3].

3.3.2. Global Vectors (GloVe)

GLoVe [61] is a type of word embedding technique that encrypts the overlapping probability ratio between two terms as vectors. GloVe seeks to minimize the difference between the dot product of the vector representations of two words and the logarithm of their number of co-occurrences using a weighted least squares objective O.

$$O = \sum_{i,k}^{V} h\left(x_{ik}\right)\left(w_i^{\mathrm{T}} \tilde{w}_k + b_i^{\mathrm{T}} + \tilde{b}_k - \log\left(x_{ik}\right)\right)^2 \tag{3.2}$$

where w_i and b_i are, respectively, the word vector and bias of word i. $\tilde{w}_k$ and $\tilde{b}_k$ are, respectively, the context word vector and bias of word k. x_{ik} represents the number of times word i appears in the context of word k. h is a function that assigns lesser weights to frequent and rare co-occurrences. We

[3] https://code.google.com/archive/p/word2vec/.

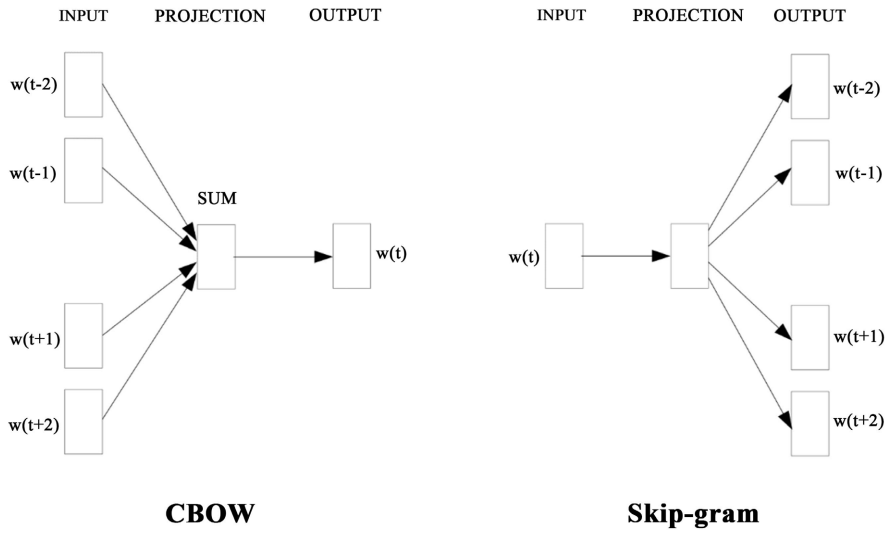


Figure 2. Word2vec model techniques.

used the glove-wiki-gigaword-300 corpus consisting of 300-dimensional vectors of 400,000 tokens obtained by training the GloVe model on Wikipedia 2014 and Gigaword 5 data and available at[4].

3.3.3. Embeddings from Language Models (ELMo)

ELMo [62] is a context-based word representation technique. ELMo word vectors are learned using a two-layer bidirectional language model (biLM) pre-trained on a large text corpus. Each layer comprises of backward and forward pass. With ELMo, a word can have several vector representations based on the context. For example, consider the following sentences.

"She kicked the bucket."

"I must accomplish every item in my bucket."

"The bucket is mine."

The word "bucket" has different meanings in each of the three sentences, hence will have distinct embeddings with ELMo in contrast to Word2vec and GloVe where it will have a single representation. We implement ELMo using the TensorFlow tools [63] which generates 1024-dimensional vector representation for each word.

3.4. Baseline Methods

State of the art bagging and boosting ensemble techniques have been chosen as baseline method for this study as well as neural network as a way to better evaluate the effectiveness of the proposed stacking approach.

3.4.1. Random Forest

Random forest is an ensemble learning method based on bagging that uses voting to combine the output of multiple decision trees. Suppose $D=\left\{\left(x_i,y_i\right),x_i\in \mathrm{R}^m,y_i\in\{0,1\}\right\}_i^n$ is the training data. For $j=1,\cdots,J$, Bootstrap

[4]https://huggingface.co/fse/glove-wiki-gigaword-300.

samples D_j of size n are created from D. For each D_j, a decision tree is fitted as describe in [64].

The output of classification for a new point x is computed as

$$\hat{f}(x) = \arg\max_{y} \sum_{j=1}^{J} I\left(\hat{g}_j(x) = y\right) \tag{3.3}$$

where $\hat{g}_j(x)$ is the prediction of the target variable at x using the j^{th} tree. The algorithm generally produces a high accuracy and is unlikely to overfit with more features. On the other hand, it also requires much time for training as it combines many decision trees.

3.4.2. Extremely Randomized Trees Classifier (Extra Trees Classifier)

Extra tree [65] is quite similar to a random forest classifier and only differs from it in the way the decision trees in the forest are built. Each decision tree is fitted to the original training sample such that, at each test node, k randomly selected features from the feature set are provided, from which it must choose the best feature to create the split according to some mathematical criteria (usually the Gini Index). This random sample of features leads to significantly lower computational time and variance compared to random forest and helps to produce multiple de-correlated trees.

3.4.3. Adaptive Boosting (Adaboost)

Let $\left\{\left(x_i, y_i\right): x_i \in X, y_i \in \left\{-1,1\right\}\right\}_i^n$ be the training data. For iteration $t = 1, \cdots, T$, The dataset is sampled using a weight distribution W_t initially given by $W_1(i) = \frac{1}{n}$ and a weak learner, Decision tree in our case, is trained repeatedly on the training set to obtain the classifiers $h_t : X \to \left\{-1,1\right\}$ with minimized weighted error ε_t relative to W_t.

$$\varepsilon_t = \Pr_{i \sim W_t}\left[f_t(x_i) \neq y_i\right] = \sum_{i: f_t(x_i) \neq y_i} W_t(i) \tag{3.4}$$

A weight parameter η_t, for each t is computed as

$$\eta_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{e_t}\right) \tag{3.5}$$

W_t is updated is such manner that incorrectly predicted instances are given more weight, while correctly predicted instances are given less weight so that more misclassified samples will be included in the next iteration. Thus, giving them more focused in the subsequent iteration and reducing error.

$$W_{t+1}(i) = \frac{W_t(i)\exp\left(-\eta_t y_i f_t(x_i)\right)}{N_t} \tag{3.6}$$

N_t is a normalization factor needed for W_t to be a distribution. The final classifier F outputs the sign of a weighted combination of weak classifiers

$$\mathrm{F}(x) = \operatorname{sign}\left\{\sum_{t=1}^{T} \eta_t f_t(x)\right\} \tag{3.7}$$

The algorithm produces low bias and variance and can yield high accuracy like other boosting algorithms. On the other hand, it also requires much time for training as it combines many weak learners.

3.4.4. Gradient Boosting Machine (GBM)

GBM is an ensemble machine learning algorithm that uses multiple Decision Trees as weak learners. This algorithm's fundamental principle is to develop models in a sequential manner, with each model building on the errors or residuals of the preceding one in an effort to reduce the errors of the prior model. GBM can be used in classification, but all base models are regression models.

Let $\{x_i, y_i\}_i^n$ be the dataset, where $x_i \in \mathbb{R}^m$ is a collection of independent variable and $y_i \in \{0,1\}$ represents the target. The initial constant prediction γ (log(odds)) are computed as to minimize the loss function L (typically the log-loss for classification) as

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma) \tag{3.8}$$

For iteration $m \in \{1, \cdots, M\}$, the residuals or errors are computed as

$$\hat{y}_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{f(x)=f_{m-1}(x)} \tag{3.9}$$

For all m, each weak learner $T_m(x)$ is fitted on the new dataset $\{x_i, \hat{y}_{im}\}_1^n$ as to minimize the loss function as

$$(\lambda_m, T_m(x_i)) = \arg\min_{\lambda, T} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \lambda T(x_i)) \tag{3.10}$$

where L is a loss function and measures the difference between the actual and predicted value. λ_m is the model's weight at iteration m. The model is updated as

$$F_m(x_i) = F_{m-1}(x_i) + \lambda_m T_m(x_i) \tag{3.11}$$

For binary classification, the probability value that a new point x belongs to the positive class is modeled as $P(y=1|x) = \sigma(F_M(x))$, where σ represents the sigmoid function and the output class is computed as

$$y = \begin{cases} 1 & \text{if } P(y=1|x) \geq 0.5 \\ 0 & \text{if } P(y=1|x) < 0.5 \end{cases} \tag{3.12}$$

GBM can be more accurate than RF and ET because each weak learner is trained in an attempt to correct the errors of the predecessor. On the other hand, it also requires much time for training as it combines many weak learners and the algorithm is likely to overfit with a noisy data.

3.4.5. Extreme Gradient Boosting (XGBoost)

The XGBoost is a novel implementation method for GBM introduced by [66]. XGBoost tries to minimize computation resources while simultaneously preventing over-fitting. This is achieved by simplifying the objective functions which consist of the regularization and predictive terms, but preserving an op-

timal computational speed. Additionally, during the training phase, parallel computing is automatically carried out for the functions in XGBoost which speed up the running time. Recently, XGBoost has outperformed other machine learning approaches in Kaggle and applied machine learning competitions as well as in a number of studies.

The additive learning process in XGBoost is described below. First, the entire space of training data is fitted to the first learner, and subsequent learners are fitted to the residuals to address the shortcomings of the prior learner. This process is repeated until the stopping criteria are met. The model's final prediction is calculated by adding each learner's predictions.

Suppose $D=\left\{\left(x_i,y_i\right)\right\}_i^n$ is the training data, where x_i and y_i respectively represents the set of descriptors and class labels. The model expression is given by

$$\hat{y}_i^{(t)}=\sum_{k=1}^{t}d_k\left(x_i\right)=\hat{y}_i^{(t-1)}+d_t\left(x_i\right) \tag{3.13}$$

where $\hat{y}_i^{(t)}$ represent the predicted value at the t^{th} iteration; x_i is the input vector; $d_t\left(x_i\right)$ is the learner at iteration t. The objective function is given by

$$\begin{aligned} L^t&=\sum_{i=1}^{n}l\left(y_i,\hat{y}_i^{(t-1)}+d_t\left(x_i\right)\right)+\Omega\left(d_t\right)\\ \Omega&=\gamma T+\frac{1}{2}\lambda\left\|w\right\|^2 \end{aligned} \tag{3.14}$$

where l is a loss function, Ω is a regularization term that avoid overfitting by penalizing the model's complexity. w and T respectively represent the score on each leaf and the number of leaves. λ and γ are constants used to regulate the level of regularization. Applying the Taylor's expansion of the loss function up to the second order gives

$$\begin{aligned} L^t&=\sum_{i=1}^{n}\left[h_id_i\left(x_i\right)+\frac{1}{2}g_id_i^2\left(x_i\right)\right]+\gamma T+\frac{1}{2}\lambda\sum_{j=1}^{T}w_j^2\\ &=\sum_{j=1}^{T}\left[\left(\sum_{i\in I_j}h_i\right)w_j+\frac{1}{2}\left(\sum_{i\in I_j}g_i+\lambda\right)w_j^2\right]+\gamma T \end{aligned} \tag{3.15}$$

where $h_i=\partial_{\hat{y}_i^{(t-1)}}l\left(y_i,\hat{y}_i^{(t-1)}\right)$ and $g_i=\partial^2_{\hat{y}_i^{(t-1)}}l\left(y_i,\hat{y}_i^{(t-1)}\right)$ are, respectively, the first and second derivative. $I_j=\left\{i:p\left(x_i\right)=j\right\}$ represents the instance set of the j^{th} leaf. This second-order expansion technique of the loss function, makes it quicker and more efficient than traditional gradient boosting algorithms.

During splitting, a leaf node is scored using Equation 3.16. The score on the left, right, and original leaf, respectively, are represented by the first, second, and third terms of the equation while the last phrase is the regularization on the additional leaf.

$$Gain=\frac{1}{2}\left[\frac{\left(\sum_{i\in I_j}h_i\right)_L^2}{\left(\sum_{i\in I_j}g_i\right)_L+\gamma}+\frac{\left(\sum_{i\in I_j}h_i\right)_R^2}{\left(\sum_{i\in I_j}g_i\right)_R+\gamma}-\frac{\left(\left(\sum_{i\in I_j}h_i\right)_L+\left(\sum_{i\in I_j}h_i\right)_R\right)^2}{\left(\sum_{i\in I_j}g_i\right)_L+\left(\sum_{i\in I_j}g_i\right)_R+\lambda}\right]-\gamma \tag{3.16}$$

XGBoost grows tree using level-wise strategy as describe in **Figure 3** which results to higher complexity compared to LightGBM.

3.4.6. Light Gradient Boosting Machine (LightGBM)

LightGBM [67] is a gradient boosting decision tree implementation. LGBM is different from other GBM implementations in that it employs the gradient-based one-side sampling (GOSS) and leaf-wise growth strategy when training each decision tree. GOSS seeks to solve the computational complexity problem with conventional GBM implementations, which must examine each attribute of each data point when calculating the information gain for all possible splits. The mechanism of GOSS is that, data instances with large gradients contribute more to the computation of information gain. As a result, GOSS chooses instances with large gradients and randomly samples instances with small gradients. This technique makes LightGBM faster and effective than conventional once.

Leaf-wise growth as describe in **Figure 4** is a strategy for growing trees. Each time, it searches through the current leaves to locate the one with the highest splitting gain, splits it, and repeats the process. In other words, it will select the leaf with the highest delta loss to grow. Compared with level-wise growth, the leaf-wise strategy can reduce much more errors and achieve better accuracy. The drawback of the leaf-wise approach is that it may grow tree deeply and results in overfitting. LightGBM tackles this by adding a maximum depth limit to ensure better efficiency while preventing overfitting. LightGBM is susceptible to overfitting and thus, easily overfit with small data.

Let $D = \left\{\left(x_i, y_i\right) : x_i \in R^m, y_i \in \left\{0,1\right\}\right\}_i^n$ be the training data. The model expression is given by

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k\left(x_i\right) = \hat{y}_i^{(t-1)} + f_t\left(x_i\right) \tag{3.17}$$

where $\hat{y}_i^{(t)}$ represent the predicted value at the t^{th} iteration; x_i is the input vector; $f_t\left(x_i\right)$ is the learner at iteration t. The objective function is given by.

$$\begin{aligned} L^t &= \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t\left(x_i\right)\right) + \Omega\left(f_t\right) \\ \Omega &= \gamma T + \frac{1}{2}\lambda \left\|w\right\|^2 \end{aligned} \tag{3.18}$$

where l is a loss function, Ω is a regularization term that avoid overfitting by penalizing the model's complexity. w and T respectively represent the score on each leaf and the number of leaves. λ and γ are constants used to regulate the level of regularization. Applying the Taylor's expansion of the loss function up to the second order gives

$$\begin{aligned} L^t &= \sum_{i=1}^{n}\left[h_i d_i\left(x_i\right) + \frac{1}{2} g_i d_i^2\left(x_i\right)\right] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \\ h_i &= \partial_{\hat{y}_i^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right) \\ g_i &= \partial^2_{\hat{y}_i^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right) \end{aligned} \tag{3.19}$$

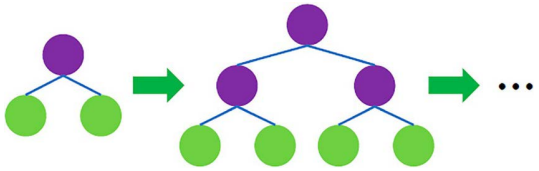


Figure 3. Level wise growth.

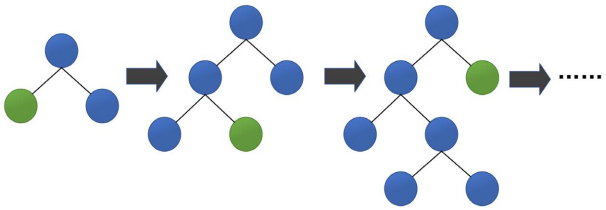


Figure 4. Leaf wise growth.

To implement GOSS, First, instances are ranked in descending order based on the magnitude of their negative gradient loss function. Let $\{g_1,\cdots,g_n\}$ be the negative gradient loss function in each iteration. Second, the data is splitted based on the absolute values of the gradient. A subset A consisiting of the top-a 100% instances with larger gradient is created. While a subset B of size $b\times|A^c|$ is sampled from the remaining samples. The instances are then divided based on the variance gain $\tilde{V}_j(q)$ over the subset $A\cup B$ as

$$\tilde{V}_j(q)=\frac{1}{n}\left(\frac{\left(\sum_{x_i\in A_l}g_i+\frac{1-a}{b}\sum_{x_i\in B_l}g_i\right)^2}{n_l^j(q)}+\frac{\left(\sum_{x_i\in A_r}g_i+\frac{1-a}{b}\sum_{x_i\in B_r}g_i\right)^2}{n_r^j(q)}\right) \quad (3.20)$$

where $A_l=\{x_i\in A_i:x_{i,j}\le q\}$, $A_r=\{x_i\in A_i:x_{i,j}>q\}$, $B_r=\{x_i\in B:x_{i,j}\le q\}$, $B_l=\{x_i\in B:x_{i,j}>q\}$.

3.4.7. Multi-Layer Perceptron (MLP)

Multi-layer perceptron is a feed forward neural network that uses backpropagation to train the weights of the network [68]. The MLP consist of an input layer, one or more hidden layer and an output layer. Each layer consist of simple building blocks call neurons. Multilayer perceptron algorithm involves repeated cycle of steps. Each cycle is call an epoch.

First, the weights and bias are initialized with random values. Let $k=0,\cdots,p$ denote the layer index and $i=1,\cdots,n_k$ denote the neuron index within layer k. Let $\{(x_u,y_u),u=1,\cdots,N\}$ be the training data. Where $y_u\in\{0,1\}$ and $x_u=\{x_{1u},\cdots,x_{qu}\}\in\mathbb{R}^q$. Every input neuron is represented by a several variable function given by:

$$\begin{aligned} f_i^{(k)}&:\mathbb{R}^q\to\mathbb{R}, \quad q=n_0\\ f_i^{(k)}(x_u)&=x_{iu}, \quad k=0 \end{aligned} \quad (3.21)$$

Every non input neuron is represented by 2 several variable functions

$a_i^{(k)}: \mathbb{R}^q \to \mathbb{R}$, $f_i^{(k)}: \mathbb{R}^q \to \mathbb{R}$. Neuron i of layer $k \neq 0$ receives input from layer $k-1$ with weights and bias represented by $w_{ij}^{(k-1)}$ and $b_i^{(k-1)}$. Its output is computed as

$$\begin{aligned} f_i^{(k)} &= \sigma\left(a_i^{(k)}\right), k \neq 0 \\ a_i^{(k)} &= \boldsymbol{\sigma}\left[\sum_{j=1}^{n_{k-1}} f_j^{(k-1)} w_{ij}^{(k-1)} + b_i^{k-1}\right] \end{aligned} \tag{3.22}$$

$\boldsymbol{\sigma}$ is an activation function. The output of the network is given by

$$f = \left(f_1^{(p)}, \cdots, f_{n_p}^{(p)}\right) = \left(\sigma\left(a_1^{(p)}\right), \cdots, \sigma\left(a_{n_p}^{(p)}\right)\right) \tag{3.23}$$

For binary classification, the final layer is a single output neuron ($n_p = 1$) with sigmoid activation function given by

$$\boldsymbol{\sigma}(x) = \frac{1}{1+\mathrm{e}^{-x}} \tag{3.24}$$

Backpropagation involves computing $\frac{\partial E}{\partial w_{ij}^{(k)}}$ and $\frac{\partial E}{\partial b_i^{(k)}}$ for all i and j. Where E is the error function and is computed as

$$E = -\frac{1}{N}\sum_{u=1}^{N}\left[y_u \log\left(f\left(x_u\right)\right) + \left(1-y_u\right)\log\left(1-f\left(x_u\right)\right)\right] \tag{3.25}$$

For a small positive number η known as the training rate, the weights and bias are adjusted as

$$\begin{aligned} w_{ij}^{(k)} &\leftarrow w_{ij}^{(k)} - \eta \frac{\partial E}{\partial w_{ij}^{(k)}} \\ b_i^{(k)} &\leftarrow b_i^{(k)} - \eta \frac{\partial E}{\partial b_i^{(k)}} \end{aligned} \tag{3.26}$$

The cycle then restarts with the updated weights and bias. When E is less than a specified number or if a certain number of specified epoch has been completed, then the algorithm terminates. The algorithm outputs the class of a new data point x as

$$y = \begin{cases} 1 & \text{if } f(x) \geq 0.5 \\ 0 & \text{if } f(x) < 0.5 \end{cases} \tag{3.27}$$

The algorithm is suitable for linear and non-linear relationships. On the other hand, it has a tendency to overfit, has poor generalization ability and is computationally expensive.

3.5. The Proposed Methodology-Stacked Ensemble Technique

Extra tree, XGBoost, and LGBM are the base learners in our suggested stacked methodology. These techniques have been chosen because they can yield optimal performance and have significantly lower training time compared to other bagging and boosting techniques. These attributes will help to minimize the computational cost of the stack while aiming to maximize performance. Multi-layer

perceptron is suitable for linear and non-linear relationships and has a weighting scheme for learning the optimal combination of base models' prediction.

The base learners are trained on the training data with a V-fold cross validation and out of fold probabilistic output of the positive class are used as explanatory variables to train the MLP with the training data target variable.

Let $\left\{\left(x_i, y_i\right), x_i \in \mathbb{R}^m, y \in \{0,1\}\right\}_i^n$ be the training data. Let $\hat{\Psi}_{ET}$, $\hat{\Psi}_{XGBoost}$, $\hat{\Psi}_{LGBM}$ be the base learners. That is, Extra Tree, XGBoost and LightGBM leaners respectively.

The training data is split into training and validation samples. Let $u = 1, \cdots, V$ be the sample split index. For all *u*, the validation samples $V(u)$ and training samples $T(u)$ satisfy the following conditions

$$T(u) = \left\{\left(x_i, y_i\right)\right\}_i^n \setminus V(u) \tag{3.28}$$

$$V(u) \cup T(u) = \left\{\left(x_i, y_i\right)\right\}_i^n \tag{3.29}$$

$$\bigcup_{u=1}^{V} V(u) = \left\{\left(x_i, y_i\right)\right\}_i^n \tag{3.30}$$

$$V(u_1) \cap V(u_2) = \varnothing, \quad u_1 \neq u_2 \tag{3.31}$$

Each base estimator is trained on $T(u)$, for each *u*, to realise

$$\Psi_{ET,u} = \hat{\Psi}_{ET}\left(T(u)\right) \tag{3.32}$$

$$\Psi_{XGBoost,u} = \hat{\Psi}_{XGBoost}\left(T(u)\right) \tag{3.33}$$

$$\Psi_{LGBM,u} = \hat{\Psi}_{LGBM}\left(T(u)\right) \tag{3.34}$$

Let $u(i)$ denote the validation sample that observation *i* belongs to, $i = 1, \cdots, n$. A new dataset $\left\{\left(z_i, y_i\right) : z_i \in \mathbb{R}^3, y_i \in \{0,1\}\right\}_i^n$ is created. Where z_i is a vector consisting of the base models' predicted probabilistic output value of the positive class for observation *i*, trained on $T\left(u(i)\right)$.

$$z_i = \left\{\Psi_{ET,u(i)}\left(y = 1/x_i\right), \Psi_{XGBoost,u(i)}\left(y = 1/x_i\right), \Psi_{LGBM,u(i)}\left(y = 1/x_i\right)\right\} \tag{3.35}$$

The MLP is then trained using of $Y = \left\{y_i\right\}_i^n$ as target and $Z = \left\{z_i\right\}_i^n$ as explanatory variables (MLP($Y|Z$)) as describe in subsection 3.4.7 to produce the stacked ensemble model. In this work, only 5-fold cross validation was implemented ($V = 5$) to limit the computational cost of training the stacked classifier. Figure 5 is a flow chart for the implementation of the stacking approach.

3.6. Performance Evaluation Metrics

The evaluation metrics allow us to test the quality of the models through their performances. The accuracy, precision, recall, F1-measure, and AUC have been used in this study to evaluate the effectiveness of the models. The average performance score of each classifier has been considered based on 10 runs. In each run, 80% of the data is sampled for training the models while the remaining in-stances are used for testing.

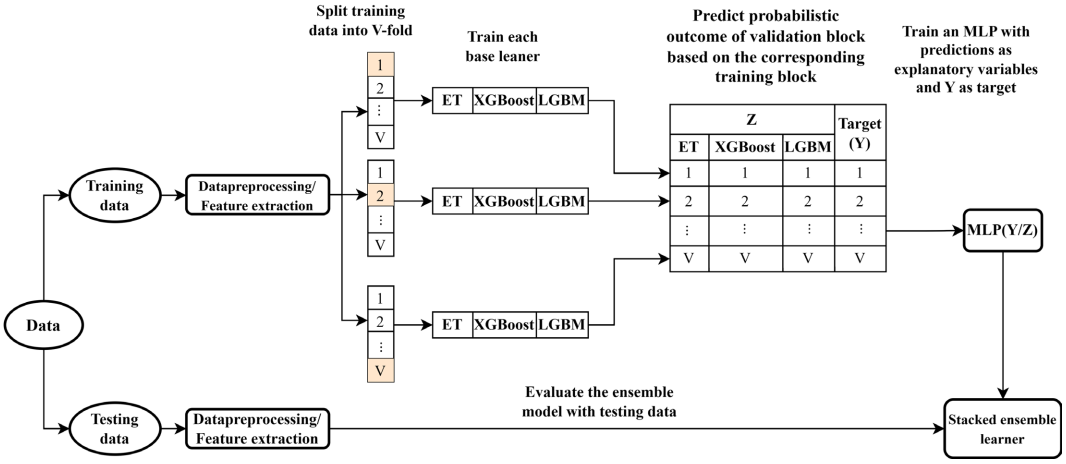


Figure 5. Flow chart of stacked ensemble approach for detecting depression from social media text.

The accuracy measures how often a model classifies instances correctly.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{3.36}$$

F1-measure is the harmonic mean between recall and precision values. Recall measure how often a model is able to classify positive instances correctly. Precision measures the proportion of correct positive predictions.

$$\text{F1-Measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.37}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.38}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.39}$$

The Receiver operating characteristic curve is a plot of true positive ratio (TPR) against false positive ratio (FPR) for different classification thresholds. The higher the area under this curve (AUC), the better the model. The AUC value reflects the overall ranking performance of a classifier.

$$\begin{aligned} \text{TPR} &= \frac{\text{TP}}{\text{TP} + \text{FN}} \\ \text{FPR} &= \frac{\text{FP}}{\text{FP} + \text{TN}} \end{aligned} \tag{3.40}$$

In the equations above, TP, TN, FP, and FN respectively represent, the number of true positives, true negatives, false positives and false negatives. True positives are correct positively predicted instances. False positives are incorrect positively predicted instances. True negatives are correct negatively predicted instances. False negatives are incorrect negatively predicted instances.

3.7. Hyperparameter Optimization

Hyperparameters are variables whose values control the learning process. Hyperparameter optimization of an algorithm involves finding the hyperparameter values that result in the most optimal performance of the learning algorithm. The

Random Search technique was used for hyperparameter optimization. The Randomized Search CV function of Scikit Learn allows us to provide lists of hyperparameter values and a random combination of values are selected from these space to train the algorithm. This technique requires less computation and execution time and was used to minimize the training time of the classifiers given a large number of features. Table 1 describes the hyperparameters of the algorithms and the search space that was implemented.

3.8. Statistical Test

To compare and analyze the performance of proposed stacked classifier to the baseline classifiers, we perform a modified version of the paired t-test proposed by Nadeau and Bengio [69] which corrects the variance estimate by taking dependencies into consideration. The paired t-test may not be a suitable choice due to the normality and independence assumption. In different runs, the training and test sets overlap and are not independent as a result. Thus, applying the paired t-test violates this assumption and the main consequence is a high type 1 error (rejecting the null hypothesis when it is true).

Let $x_1^s, \cdots, x_n^s$ be the observed prediction scores from the stacked classifier and $x_1^b, \cdots, x_n^b$ be the observed scores from a baseline classifier, where n is the number of runs. Because we want to know if the stacked ensemble classifier will outperform the baseline classifier in terms of accuracy, the null and alternative hypothesis are respectively constructed as

$$\begin{aligned} H_0 &: z_j = x_j^s - x_j^b = 0 \\ H_1 &: z_j = x_j^s - x_j^b > 0 \end{aligned} \tag{3.41}$$

The modified t-statistics is computed as

$$t = \frac{\frac{1}{n}\sum_{j=1}^{n} z_j}{\sqrt{\left(\frac{1}{n} + \frac{n_1}{n_2}\right)\frac{1}{n-1}\sum_{j=1}^{n}\left(z_j - \bar{z}\right)^2}} \tag{3.42}$$

where n_1 and n_2 represents the number training and test samples respectively. The experiment was implemented with 10 runs ($n=10$) and thus, a 9 ($n-1$) degree of freedom. The significant level is $\alpha = 0.05$ and thus, the null hypothesis is rejected if $t > 1.83$ or if p-value < 0.05. Limiting the number of runs to 10 aimed at minimizing the computational cost of training the models given the relatively large number of features.

4. Results and Discussion

To evaluate the effectiveness of the proposed stacked model, the experimental results of the model are compared to the baseline models. Each of the classifiers was implemented on the two datasets with 10 runs. The section presents, analyzes and compares the means performance scores of the stacked classifier to the baseline classifiers after implementation with the three feature extraction

Table 1. Hyperparameters, descriptions and search space.

Classifier	Hyper-parameter name	Description	Optimized value space
MLP	hidden_layer_sizes	Number of hidden layers and neurons	[100, 70, 50]
	Activation	Output function of non-input neuron	[logistic, relu, tanh]
	learning_rate	Regularization parameter	[0.1, 0.001, 0.0001]
	Alpha	Controls step size during weight updates	[0.0001, 0.01, 0.1]
	Solver	Weight optimization method	[lbfgs, sgd, adam]
RF ET	n_estimators	Number of trees in the forest	[70, 100, 120]
	max_depth	Maximum depth of the tree.	[3, 5, 8]
	min_samples_leaf	Minimum number of samples at a leaf node.	RF = [2, 5, 10], ET = [1, 2, 5]
	Criterion	Measure the quality of a split	[gini, entropy]
GBM	n_estimators	Number of tress.	[70, 100, 120]
	learning_rate	Shrinkage factor for each tree	[0.1, 0.001]
	min_samples_leaf	Minimum number of samples needed to be at a leaf node	[5, 10]
	max_depth	Maximum depth of individual tree	[3, 5, 7]
	Loss	The loss function in the boosting process to be optimized.	[log_loss, exponential]
Adaboost	n_estimators	Number of trees.	[70, 100, 120]
	learning_rate	Boosting learning rate	[0.1, 0.001]
XGBoost	n_estimators	Number of trees.	[70, 100, 120]
	learning_rate	Shrinkage factor of each tree	[0.1, 0.001]
	max_depth	Tree depth	[5, 10, 20]
	subsample	Subsample ratio of training samples	[0.1, 0.5, 1]
LightGBM	n_estimators	Number of gradient boosted trees.	[70, 100, 120]
	learning_rate	Shrinkage factor of each tree	[0.1, 0.001]
	max_depth	Maximum tree depth for base learners.	[10, 20, 30]
	num_leaves	Number of leaves for each tree	[5, 8, 15]

techniques as well as their execution time. The results of the statistical test is presented and analyzed. The section also compares the best results of the stacked classifiers to recent works in the literature.

4.1. Results

The subsection presents the results of the methods on the datasets. The mean test performance scores from the 10 runs experiment are displayed in Table 2 and Table 3. Table 4 and Table 5 compare the performance differences between the stack and the baseline techniques. In the tables, acc represents mean accuracy, pre represents the mean precision, rec represents the mean recall, f1 represents the mean F1-score and auc represents the mean AUC. The highest

Table 2. Mean scores of classifiers for depression detection based on 10-runs experiment on Sentiment_tweets3 dataset.

Method	Word2vec					GloVe					ELMo				
	acc	pre	rec	f1	auc	acc	pre	rec	f1	auc	acc	pre	rec	f1	auc
RF	94.34	**99.25**	85.63	91.94	98.79	92.55	93.31	81.25	86.85	97.04	96.75	98.87	92.17	95.41	99.16
ET	98.71	98.78	97.41	98.09	99.45	91.75	92.00	85.03	88.22	96.72	98.72	98.45	97.11	97.77	99.88
AdaBoost	96.24	96.61	93.99	95.28	99.17	87.77	86.59	85.41	85.99	96.04	98.17	96.60	95.89	96.24	99.38
GBM	98.10	98.18	93.26	95.65	99.53	93.20	88.35	80.34	84.13	97.48	97.46	97.78	90.75	94.12	99.54
XGBoost	98.13	98.67	95.70	97.16	99.75	92.78	90.73	89.62	90.03	**98.06**	98.60	99.05	96.82	97.92	99.63
LightGBM	98.71	98.75	97.66	98.20	99.38	92.80	91.29	89.62	89.82	97.07	99.08	98.52	97.93	98.22	99.60
MLP	98.79	98.26	97.41	97.83	99.69	93.01	85.97	85.26	84.97	97.89	99.09	98.84	**98.58**	98.70	99.95
STACK	**98.99**	99.14	**97.98**	**98.56**	**99.93**	**94.47**	**94.48**	**91.29**	**92.85**	98.05	**99.40**	**99.70**	98.44	**99.05**	**99.96**

Table 3. Mean scores of classifiers for depression detection based on 10-runs experiment on depression_dataset_reddit_cleaned dataset.

Method	Word2vec					GloVe					ELMo				
	acc	pre	rec	f1	auc	acc	pre	rec	f1	auc	acc	pre	rec	f1	auc
RF	93.12	**97.59**	87.91	92.47	98.20	91.55	**97.24**	85.38	90.85	97.71	96.75	99.57	89.40	92.18	99.50
ET	93.95	93.85	93.94	93.89	97.91	92.20	91.85	92.48	92.15	96.97	98.94	98.96	98.90	98.92	99.96
AdaBoost	90.60	90.79	90.76	90.76	96.67	90.53	90.50	90.55	90.50	96.57	96.97	97.45	96.42	96.85	99.61
GBM	92.73	94.52	90.58	92.49	97.66	91.70	94.01	88.91	91.36	97.28	95.27	97.81	92.56	94.68	99.16
XGBoost	94.68	95.91	92.64	94.21	98.59	93.91	95.73	91.80	93.71	98.34	97.70	99.18	96.19	97.54	99.85
LightGBM	94.60	95.21	94.08	94.62	98.08	92.80	92.18	92.25	92.20	97.11	99.52	99.35	99.69	99.52	99.92
MLP	94.33	94.25	94.36	94.28	98.31	93.64	94.04	**93.58**	93.77	97.90	99.27	**99.84**	99.09	99.46	99.93
STACK	**95.59**	96.51	**94.41**	**95.43**	**98.62**	**94.01**	94.32	93.45	**93.86**	**98.35**	**99.54**	99.45	**99.84**	**99.65**	**99.98**

Table 4. Mean performance differences between the stack approach and the baseline models on Sentiment_tweets3 dataset.

Method	Word2vec					GloVe					ELMo				
	acc	pre	rec	f1	auc	acc	pre	rec	f1	auc	acc	pre	rec	f1	auc
RF	4.65	**0.11**	12.35	6.62	1.14	1.92	**1.17**	10.04	6	1.01	2.65	0.83	6.27	3.64	0.8
ET	0.28	0.36	0.59	0.47	0.48	2.72	2.48	6.26	4.63	1.33	0.68	1.25	1.33	1.28	0.08
AdaBoost	2.75	2.53	3.99	3.28	0.76	6.7	7.89	5.88	6.86	2.01	1.23	3.1	2.55	2.81	0.58
GBM	0.89	0.96	4.72	2.91	0.4	**1.27**	6.13	10.95	8.72	0.57	1.94	1.92	7.69	4.93	0.42
XGBoost	0.86	0.47	2.28	1.4	**0.18**	1.69	3.75	**1.67**	**2.82**	**0.01**	0.8	**0.65**	1.62	1.13	0.33
LightGBM	0.28	0.39	0.72	**0.36**	0.55	**1.67**	3.19	1.67	3.03	0.98	0.32	1.18	0.51	0.83	0.36
MLP	**0.2**	0.88	**0.57**	0.73	0.24	1.46	8.51	6.03	7.88	0.16	**0.31**	0.86	**0.1**	**0.35**	**0.01**

Table 5. Mean performance differences between the stack approach and the baseline models on depression_dataset_reddit_cleaned dataset.

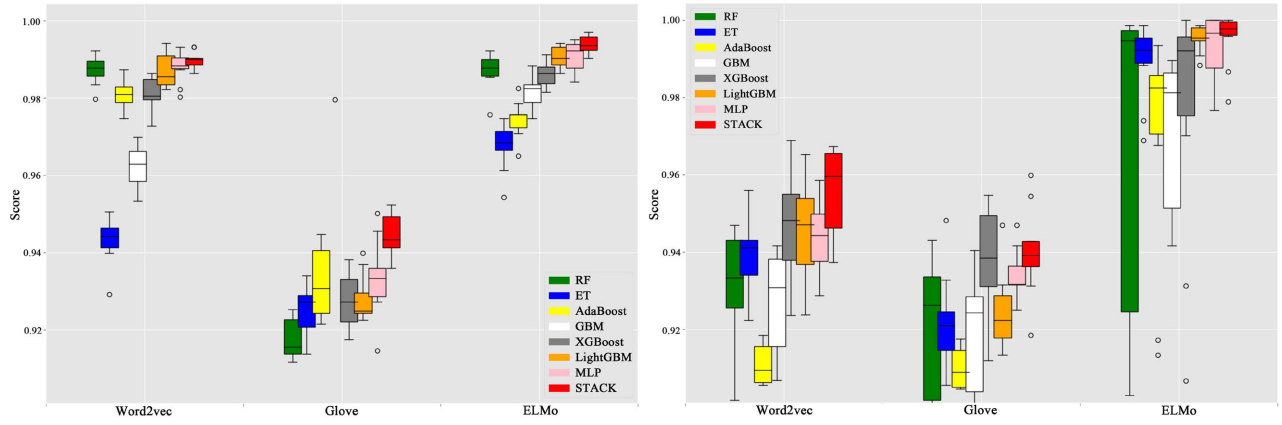
Method	Word2vec					GloVe					ELMo				
	acc	pre	rec	f1	auc	acc	pre	rec	f1	auc	acc	pre	rec	f1	auc
RF	2.47	1.08	6.5	2.96	0.42	2.46	2.92	8.07	3.11	0.64	2.79	0.12	10.44	7.47	99.50
ET	1.64	2.66	0.47	1.54	0.71	1.81	2.47	0.97	1.81	1.38	0.6	0.49	0.94	0.73	0.02
AdaBoost	4.99	5.72	3.65	4.67	1.95	3.48	3.82	2.9	3.46	1.78	2.57	2	3.42	2.8	0.37
GBM	2.86	1.99	3.85	2.94	0.96	2.31	0.31	4.54	2.6	1.07	4.27	1.64	7.28	4.97	0.82
XGBoost	**0.91**	**0.6**	1.77	1.22	**0.03**	**0.1**	1.41	1.65	0.25	**0.01**	1.84	0.27	3.65	2.11	0.13
LightGBM	0.99	1.3	0.33	**0.81**	0.54	1.21	2.14	1.2	1.76	1.24	**0.02**	**0.1**	**0.15**	**0.13**	0.06
MLP	1.26	2.26	**0.05**	1.15	0.31	0.37	**0.28**	**0.13**	**0.19**	0.45	0.27	0.39	0.75	0.19	**0.05**

performance and the smallest differences are displayed in bold. Boxplots are used in Figures 6-10 to visualize the spread and range of the test scores of each classifier from the 10 runs experiment. Table 6 and Table 7 show the *p*-values obtained from the t-test. A *p*-value less than 0.05 depicts that the accuracy increase made by stacked classifier is statistically significant at 5% level. p-Values greater than 0.05 are displayed in bold. Figure 11 shows the classifiers' execution time base on all 10 runs. Table 8 compares the best performance of this work to recent works in the literature. Best performances are shown in bold. "-" indicates the absence of the corresponding technique in the study.

4.2. Discussion

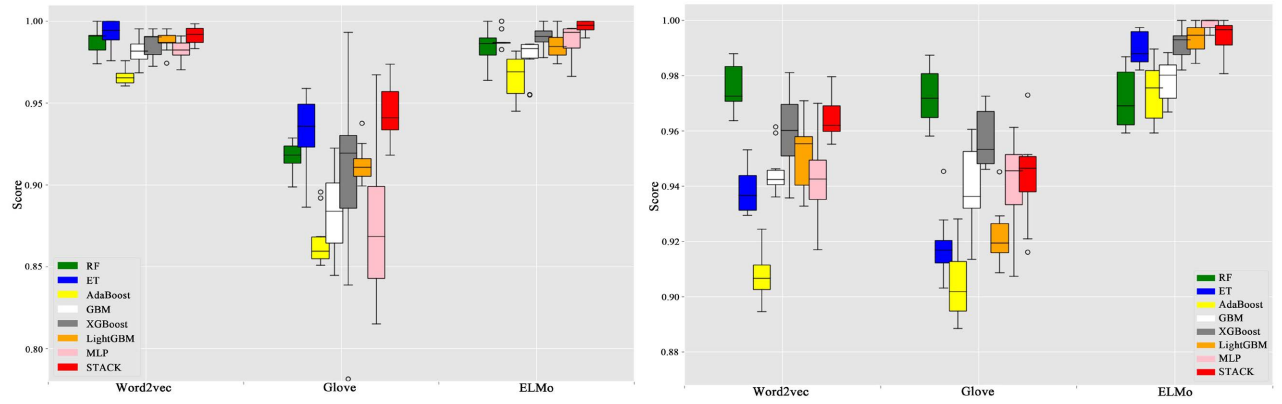
This subsection discusses the overall results of the experiments in the preceding subsection. Except for mean accuracy and F1-score, the stacked classifier did not achieve the best performance in all test cases. The accuracy measure alone may not be a suitable choice as it is biased to minority class instances. So, it is important to know which situations to prioritize any given evaluation metric in medical diagnosis. For example, the priority will be to decrease false negatives if it is most important to not misclassify individuals having the disease in order to prevent further complications. In this case, recall should be prioritized. Conversely, it may be more suitable to prioritize decrease in false positives in situations where the cost of treatment of the disease is very high. Thus, precision should be prioritized. While F1-score is prioritized when there is the need to balance between precision and recall.

The results on Sentiment_tweets3 dataset as displayed in Table 2 show that: With Word2vec, the stacked classifier achieved the best acc, rec, f1 and auc of 98.99%, 97.98%, 98.56% and 99.93% respectively while random forest classifier achieved the best pre of 99.25%. With GLoVe, the stacked classifier achieved the best acc, pre, rec and f1 of 94.47%, 94.48%, 91.29% and 92.85% respectively while the XGBoost classifier achieved the best auc of 98.06%. With ELMo, the stacked classifier achieved the best acc, pre, f1 and auc of 99.40%, 99.70%, 99.05%



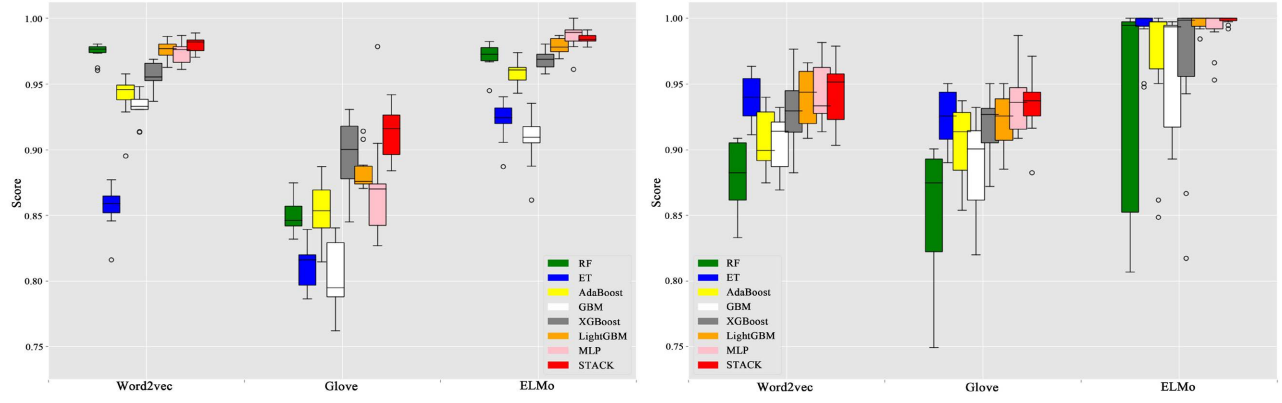
(a) Sentiment_tweets3 (b) Depression_dataset_reddit_cleaned

Figure 6. Boxplot of accuracy scores from 10-runs experiment.



(a) Sentiment_tweets3 (b) Depression_dataset_reddit_cleaned

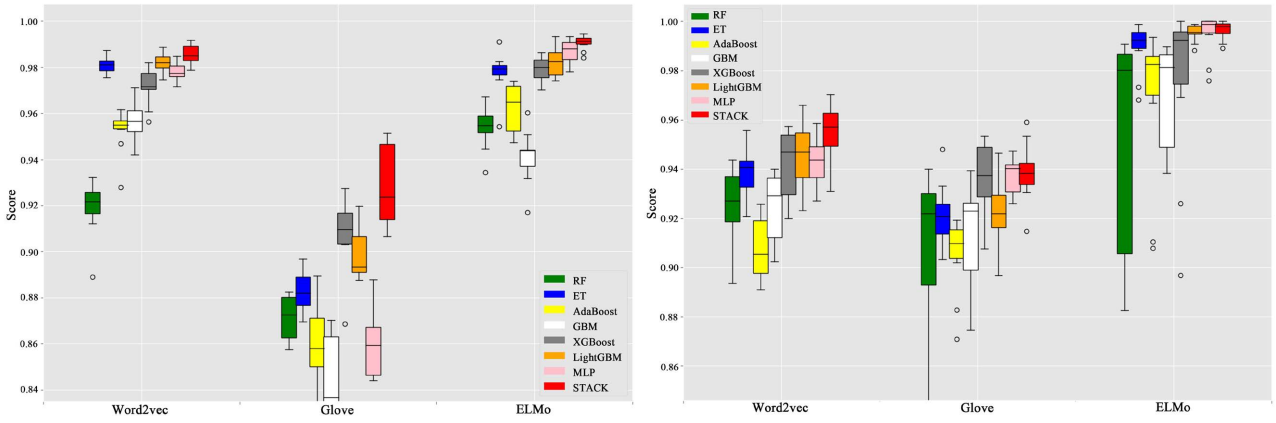
Figure 7. Boxplot of precision scores from 10-runs experiment.



(a) Sentiment_tweets3 (b) Depression_dataset_reddit_cleaned

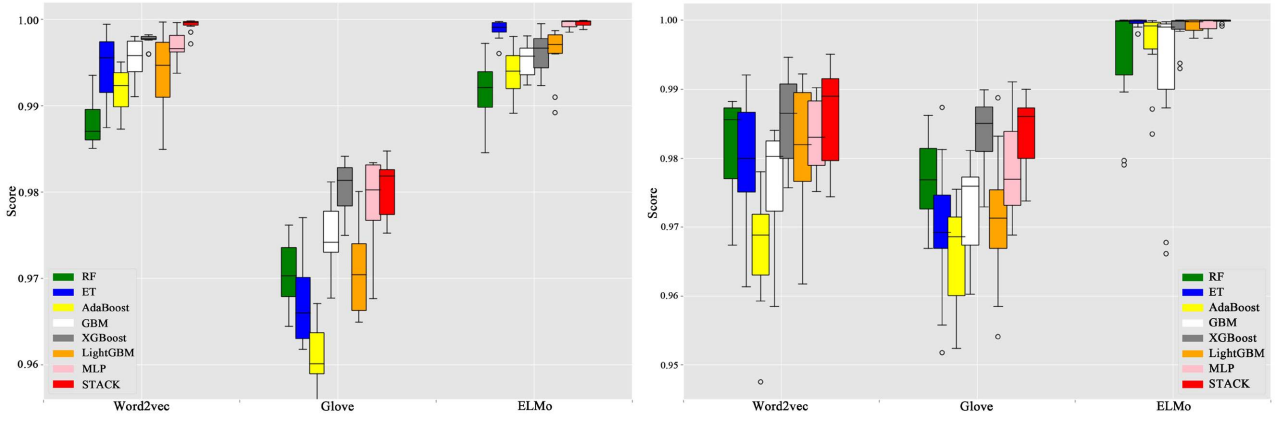
Figure 8. Boxplot of recall scores from 10-runs experiment.

and 99.96% respectively while the MLP classifier achieved the best rec of 98.58%. The results of the statistical test as displayed in **Table 6** show that the stacked classifier did not achieve a significant improvement in accuracy against LGBM with Word2vec embeddings and MLP with Word2vec and ELMo embeddings.



(a) Sentiment_tweets3 (b) Depression_dataset_reddit_cleaned

Figure 9. Boxplot of F1-scores from 10-runs experiment.



(a) Sentiment_tweets3 (b) Depression_dataset_reddit_cleaned

Figure 10. Boxplot of AUC scores from 10-runs experiment.

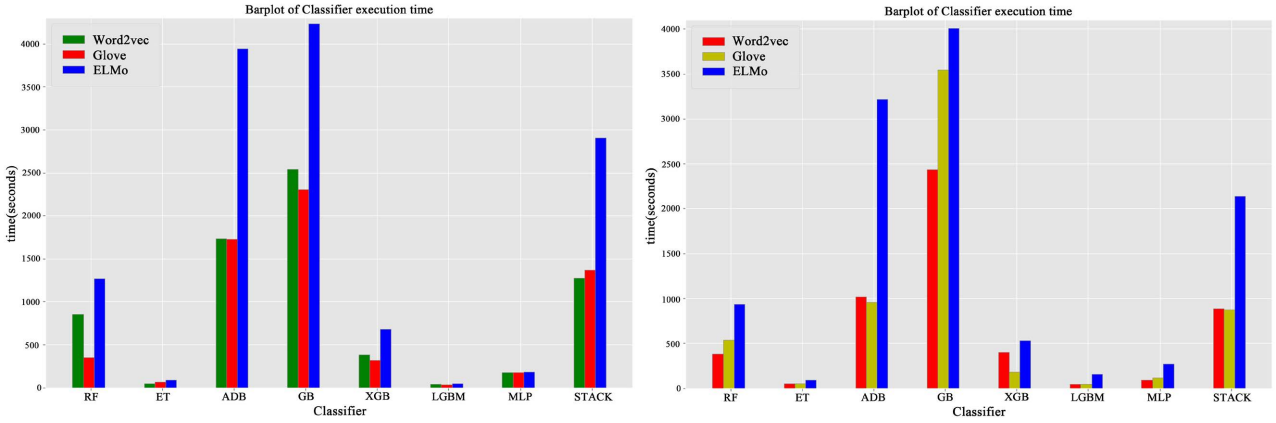
Table 6. p-Value obtained from the t-test on Sentiment_tweets3 dataset.

Method	RF	ET	AdaBoost	GBM	XGBoost	LightGBM	MLP
Word2vec	0.000	0.032	0.000	0.000	0.000	**0.144**	**0.146**
GloVe	0.000	0.000	0.001	0.000	0.002	0.000	0.044
ELMo	0.000	0.003	0.000	0.000	0.001	0.025	**0.12**

Table 7. p-Value obtained from t-test on Depression_dataset_reddit_cleaned dataset.

Method	RF	ET	AdaBoost	GBM	XGBoost	LightGBM	MLP
Word2vec	0.001	0.000	0.000	0.000	0.015	0.021	0.008
GloVe	0.030	0.001	0.000	0.011	**0.35**	0.000	**0.057**
ELMo	**0.095**	0.004	0.016	0.046	**0.072**	**0.47**	**0.16**

The results on depression_dataset_reddit_cleaned dataset as displayed in 3 show that: With Word2vec, the stacked classifier achieved the best acc, rec, f1 and auc of 95.59%, 94.41%, 95.43% and 98.62% respectively. While RF achieved



(a) Sentiment_tweets3

(b) Depression_dataset_reddit_cleaned

Figure 11. Barplot of classifiers' execution time.

Table 8. Comparison to recent works in the literature.

Study	Social Media	Feature extraction	Classifier	Accusion	Precision	reccal	F1 score	AUC	Statistical test
[36]	Twitter/Facebook	linguistic features	RF	60.54	58	60.5	54.7	-	-
[37]	Twitter	Word2vec	CNN-LSTM	94.28	96.99	92.66	94.78	95.43	t-test
[38]	Twitter/Reddit	GloVe & Sentiments	Averaging	75.12	81.15	75.12	77.01	-	-
[39]	Reddit	Word2vec	RF	87.7	-	-	87.7	-	-
[40]	Bengali S.M	word embeddings	GRU	81	81	81	81	-	-
[41]	Twitter	BERT	BERT	96.06	94.88	96.86	95.86	96.11	-
[42]	Twitter	word embeddings	WOA-CNN	93.03	90.76	92.89	91.82	-	-
[43]	Twitter	TF-IDF/LDA	XGBoost	87	86	87	87	-	-
[44]	Twitter	GloVe	CNN	93.7	92.9	94.1	93.3	-	-
[45]	Twitter	BoW	Blending	87.21	-	-	-	-	-
[46]	Twitter	BoW	Bagging	98.33	90.39	96.45	92.15	-	-
[47]	Twitter	TF-IDF	SVM	79.90	-	-	-	-	-
[48]	Twitter	Fasttext	BiLSTM + CNN	**99.74**	99.43	**99.88**	99.21	99.1	-
[49]	Twitter	LDA	CBPT	88.39	-	-	86.90	-	-
[50]	Reddit	BERT	CNN	0.86	0.87	0.85	86	-	-
[51]	Twitter	TF-IDF	RF	95	99	94	96	-	-
[52]	Facebook & Youtube	TF-IDF	SVM	75.15	77	80	78	-	-
[53]	Twitter	Word2vec	LSTM	92.89	0.88	0.60	71	-	-
This work	Reddit	ELMo	STACK	99.54	**99.45**	99.84	**99.65**	**99.98**	a corrected t-test

the best pre of 97.59%. With GLoVe, the stacked classifier achieved the best acc, f1 and auc of 94.01%, 93.86% and 98.35% respectively. The RF achieved the highest pre of 97.24% while the MLP classifier achieved the highest rec of 93.58%. With ELMo, the stacked classifier achieved the best acc, rec, f1 and auc

of 99.54%, 99.84%, 99.65% and 99.98% respectively, while the MLP classifier scored the highest pre of 99.84%. The results of the statistical test as displayed in Table 7 shows that the stacked classifier did not achieve a significant improvement in accuracy against RF, XGBoost, LGBM and MLP when implemented with ELMo. And XGBoost and MLP with GLoVe.

It is apparent from Figures 6-10 that the models generally performed best with ELMo embeddings and worst with GloVe embeddings on both datasets. This may be due to the fact that the Word2vec and GloVe embedding techniques disregard the order in which the words appear in a document, which results in a lack of syntactic and semantic comprehension of phrases to a greater extent. Whereas, ELMo takes into account, the order in which words occur. Figure 11 shows that the classifiers achieved the highest training time on both datasets with ELMo. This is due to the 1048 features of ELMo as opposed to the 300 features for Word2vec and GLoVe embeddings. The relatively high training time of the models was due to the 10-run experiment that was used in the implementations. The GBM classifier scored the highest computation time with all word representation cases, followed by the Adaboost classifier.

The results depict in general that the stacked classifier achieved the best results with ELMo embeddings on the depression_dataset_reddit_cleaned dataset even though it scored a lower mean precision to the MLP and RF classifiers. Thus, maynot be the best choice of classifier in situations where the cost of treatment of depression is very high, where it is preferable to give priority to reducing false positives. On the other hand, a highest mean recall of the stacked makes it the best classifier in cases where it is better to prioritize decrease in false negatives, while a highest mean F1-score makes it a preferable choice when it is necessary to balance between precision and recall. The improved performance of the stacked is due to the meta learner which learnt the optimal combination of its base models' predictions through weighting and error minimization. Also, in some cases, the short comings of some base classifiers were remedied by the other classifiers of the model. The stacked classifier with ELMo scored a relatively higher training time of 2138.6 seconds. Thus, other classifiers with lower training time like the LightGBM with ELMo that closely follows it in terms of performance as depicted in Table 5 can be preferable choices in situations where priority is given to lower diagnosis time, like in cases involving large amount of data to diagnose. Table 8 shows that this work achieved the best precision, F1-score and AUC compared to recent studies in the literature.

5. Conclusions

Depression is a major public health problem worldwide. Diagnosis of depression is mostly based on interviews and questionnaires. This makes the process very costly and time-consuming. Also, the disease often remains undiagnosed and suffering continues.

In this study, we implemented stacking on two datasets with three feature ex-

traction approaches for detecting depression. We evaluated the effectiveness of the stacking approach by comparing its performances to state-of-the-art boosting and bagging ensemble approaches as well as neural networks and observed an improvement in accuracy and F1-score in all test cases. We also applied a corrected resampled paired t-test to assess the significance of the accuracy improvement made by the stacking approach and found that, there was a significance at the 5% level in many cases. Finally, this work showed favorable results compared to recent works in the literature. We believe that this work can contribute to the detection of depression from social media text, in order to enable the appropriate authorities to take necessary measures to prevent subsequent complications as a result of the illness.

For further studies, other computational techniques such as deep learning can be implemented with the same feature extraction techniques, datasets and conditions used in this work and compare the findings to that of this work. The performance obtained in this study can be improved by applying other contextualized embeddings like BERT. Further, the TF-IDF weighted averaging technique can be used to capture the degree of importance of each word in every post. Lastly, this work can be extended to datasets of different languages.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] World Health Organization (2004) Promoting Mental Health: Concepts, Emerging Evidence, Practice: Summary Report. 10-12.

[2] Üstün, T.B., Ayuso-Mateos, J.L., Chatterji, S., Mathers, C. and Murray, C.J.L. (2004) Global Burden of Depressive Disorders in the Year 2000. *The British Journal of Psychiatry*, **184**, 386-392. https://doi.org/10.1192/bjp.184.5.386

[3] Mathers, C.D. and Loncar, D. (2006) Projections of Global Mortality and Burden of Disease from 2002 to 2030. *PLOS Medicine*, **3**, e442. https://doi.org/10.1371/journal.pmed.0030442

[4] Prince, M., Patel, V., Saxena, S., Maj, M., Maselko, J., Phillips, M.R. and Rahman, A. (2007) No Health without Mental Health. *The Lancet*, **370**, 859-877. https://doi.org/10.1016/S0140-6736(07)61238-0

[5] Sau, A. and Bhakta, I. (2019) Screening of Anxiety and Depression among Seafarers Using Machine Learning Technology. *Informatics in Medicine Unlocked*, **16**, Article ID: 100228. https://doi.org/10.1016/j.imu.2019.100228

[6] Chauvet-Gelinier, J.C. and Bonin, B. (2017) Stress, Anxiety and Depression in Heart Disease Patients: A Major Challenge for Cardiac Rehabilitation. *Annals of Physical and Rehabilitation Medicine*, **60**, 6-12. https://doi.org/10.1016/j.rehab.2016.09.002

[7] Ul Hassan, A., Hussain, J., Hussain, M., Sadiq, M. and Lee, S.Y. (2017) Sentiment Analysis of Social Networking Sites (SNS) Data Using Machine Learning Approach for the Measurement of Depression. 2017 *International Conference on Information and Communication Technology Convergence* (*ICTC*), Jeju, 18-20 October 2017, 138-140. https://doi.org/10.1109/ICTC.2017.8190959

[8] Nguyen, D.K., Chan, C.L., Li, A.H.A. and Phan, D.V. (2021) Deep Stacked Generalization Ensemble Learning Models in Early Diagnosis of Depression Illness from Wearable Devices Data. *Proceedings of the* 5*th International Conference on Medical and Health Informatics*, Kyoto, 14-16 May 2021, 7-12. https://doi.org/10.1145/3472813.3472815

[9] Tsugawa, S., Kikuchi, Y., Kishino, F., Nakajima, K., Itoh, Y. and Ohsaki, H. (2015) Recognizing Depression from Twitter Activity. *Proceedings of the* 33*rd Annual ACM Conference on Human Factors in Computing Systems*, Seoul, 18-23 April 2015, 3187-3196. https://doi.org/10.1145/2702123.2702280

[10] Tama, B.A. and Lim, S. (2021) Ensemble Learning for Intrusion Detection Systems: A Systematic Mapping Study and Cross-Benchmark Evaluation. *Computer Science Review*, **39**, Article ID: 100357. https://doi.org/10.1016/j.cosrev.2020.100357

[11] Gomes, H.M., Barddal, J.P., Enembreck, F. and Bifet, A. (2017) A Survey on Ensemble Learning for Data Stream Classification. *ACM Computing Surveys* (*CSUR*), **50**, 1-36. https://doi.org/10.1145/3054925

[12] Zhang, Y.Z., Liu, J.J. and Shen, W.J. (2022) A Review of Ensemble Learning aLgorithms Used in Remote Sensing Applications. *Applied Sciences*, **12**, Article 8654. https://doi.org/10.3390/app12178654

[13] Breiman, L. (1996) Bagging Predictors. *Machine Learning*, **24**, 123-140. https://doi.org/10.1007/BF00058655

[14] Freund, Y., Schapire, R.E., *et al.* (1996) Experiments with a New Boosting algorithm. In: Saitta, L., Ed., *Machine Learning*: *Proceedings of the Thirteenth International Conference* (*ICML*'96), Springer, Berlin, 148-156.

[15] Wolpert, D.H. (1992) Stacked Generalization. *Neural Networks*, **5**, 241-259. https://doi.org/10.1016/S0893-6080(05)80023-1

[16] Ma, Z.Y., Wang, P., Gao, Z.H., Wang, R.B. and Khalighi, K. (2018) Ensemble of Machine Learning Algorithms Using the Stacked Generalization Approach to Estimate the Warfarin Dose. *PLOS ONE*, **13**, e0205872. https://doi.org/10.1371/journal.pone.0205872

[17] Boehmke, B. and Greenwell, B. (2019) Hands-on Machine Learning with R. Chapman and Hall/CRC, New York. https://doi.org/10.1201/9780367816377

[18] Derczynski, L., Ritter, A., Clark, S. and Bontcheva, K. (2013) Twitter Part-of-Speech Tagging for All: Overcoming Sparse and Noisy Data. *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP* 2013, Hissar, 9-11 September 2013, 198-206.

[19] Islam, M.R., RaihanMKamal, A., Sultana, N., Islam, R., Ali Moni, M., *et al.* (2018) Detecting Depression Using K-Nearest Neighbors (KNN) Classification Technique. 2018 *International Conference on Computer*, *Communication*, *Chemical*, *Material and Electronic Engineering* (*IC4ME2*), Rajshahi, 8-9 February 2018, 1-4. https://doi.org/10.1109/IC4ME2.2018.8465641

[20] Islam, M.R., Kabir, M.A., Ahmed, A., Kamal, A.R.M., Wang, H. and Ulhaq, A. (2018) Depression Detection from Social Network Data Using Machine Learning Techniques. *Health Information Science and Systems*, **6**, Article No. 8. https://doi.org/10.1007/s13755-018-0046-0

[21] Stankevich, M., Isakov, V., Devyatkin, D. and Smirnov, I.V. (2018) Feature Engineering for Depression Detection in Social Media. *Proceedings of the* 7*th International Conference on Pattern Recognition Applications and Methods*, Funchal, 16-18 January 2018, 426-431. https://doi.org/10.5220/0006598604260431

[22] Chen, X.T., Sykora, M.D., Jackson, T.W. and Elayan, S. (2018) What about Mood

Swings: Identifying Depression on Twitter with Temporal Measures of Emotions. *Companion Proceedings of the The Web Conference* 2018, Lyon, 23-27 April 2018, 1653-1660. https://doi.org/10.1145/3184558.3191624

[23] Adegoke, A. (2019) Detection of Depression among Nigerians Using Machine Learning Techniques. Ph.D. Thesis, National College of Ireland, Dublin.

[24] Kumar, A., Sharma, A. and Arora, A. (2019) Anxious Depression Prediction in Real-Time Social Data. arXiv: 1903.10222. https://doi.org/10.2139/ssrn.3383359

[25] Kaibi, I., Satori, H. and Nfaoui, E.H. (2019) A Comparative Evaluation of Word Embeddings Techniques for Twitter Sentiment Analysis. 2019 *International Conference on Wireless Technologies*, *Embedded and Intelligent Systems* (*WITS*), Fez, 3-4 April 2019, 1-4. https://doi.org/10.1109/WITS.2019.8723864

[26] Tadesse, M.M., Lin, H.F., Xu, B. and Yang, L. (2019) Detection of Depression-Related Posts in Reddit Social Media Forum. *IEEE Access*, **7**, 44883-44893. https://doi.org/10.1109/ACCESS.2019.2909180

[27] Hemanthkumar, M. and Latha, A. (2019) Depression Detection with Sentiment Analysis of Tweets. *International Research Journal of Engineering and Technology*, **6**, 3-7.

[28] AlSagri, H.S. and Ykhlef, M. (2020) Machine Learning-Based Approach for Depression Detection in Twitter Using Content and Activity Features. *IEICE Transactions on Information and Systems*, **103**, 1825-1832. https://doi.org/10.1587/transinf.2020EDP7023

[29] Geetha, G., Saranya, G., Chakrapani, K., Godwin Ponsam, J., Safa, M. and Karpagaselvi, S. (2020) Early Detection of Depression from Social Media Data Using Machine Learning Algorithms. 2020 *International Conference on Power*, *Energy*, *Control and Transmission Systems* (*ICPECTS*), Chennai, 10-11 December 2020, 1-6. https://doi.org/10.1109/ICPECTS49113.2020.9336974

[30] Lin, C.H., Hu, P.W., Su, H., Li, S.C., Mei, J., Zhou, J. and Leung, H. (2020) Sensemood: Depression Detection on Social Media. *Proceedings of the* 2020 *International Conference on Multimedia Retrieval*, Dublin, 8-11 June 2020, 407-411.

[31] Kim, J., Lee, J., Park, E. and Han, J.Y. (2020) A Deep Learning Model for Detecting Mental Illness from User Content on Social Media. *Scientific Reports*, **10**, Article No. 11846, 2020. https://doi.org/10.1038/s41598-020-68764-y

[32] Shetty, N.P., Muniyal, B., Anand, A., Kumar, S. and Prabhu, S. (2020) Predicting Depression Using Deep Learning and Ensemble Algorithms on Raw Twitter Data. *International Journal of Electrical and Computer Engineering*, **10**, 3751-3756. https://doi.org/10.11591/ijece.v10i4.pp3751-3756

[33] Hasan Khan, M.R., Afroz, U.S., Mohammad Masum, A.K., Abujar, S. and Hossain, S.A. (2020) Sentiment Analysis from Bengali Depression Dataset Using Machine Learning. 2020 11*th International Conference on Computing*, *Communication and Networking Technologies* (*ICCCNT*), 1-3 July 2020, Kharagpur, 1-5.

[34] Rajaraman, P.V., Nath, A., Akshaya, P.R. and Chatur Bhuja, G. (2020) Depression Detection of Tweets and a Comparative Test. *International Journal of Engineering Research*, **9**, 422-425. https://doi.org/10.17577/IJERTV9IS030270

[35] Shah, F.M., Ahmed, F., Saha Joy, S.K., Ahmed, S., Sadek, S., Shil, R. and Kabir, M.H. (2020) Early Depression Detection from Social Network Using Deep Learning Techniques. 2020 *IEEE Region* 10 *Symposium* (*TENSYMP*), Dhaka, 5-7 June 2020, 823-826.

[36] Saha, A., Al Marouf, A. and Hossain, R. (2021) Sentiment Analysis from Depressionrelated User-Generated Contents from Social Media. 2021 8*th International Con-*

ference on Computer and Communication Engineering (*ICCCE*), Kuala Lumpur, 22-23 June 2021, 259-264. https://doi.org/10.1109/ICCCE50029.2021.9467214

[37] Dabhane, S. and Chawan, P.M. (2020) Depression Detection on Social Media Using Machine Learning Techniques. *International Research Journal of Engineering and Technology*, **7**, 97-100.

[38] Liu, J.F. and Shi, M.S. (2022) A Hybrid Feature Selection and Ensemble Approach to Identify Depressed Users in Online Social Media. *Frontiers in Psychology*, **12**, Article 802821. https://doi.org/10.3389/fpsyg.2021.802821

[39] Kour, H. and Gupta, M.K. (2022) An Hybrid Deep Learning Approach for Depression Prediction from User Tweets Using Feature-Rich CNN and Bi-Directional LSTM. *Multimedia Tools and Applications*, **81**, 23649-23685. https://doi.org/10.1007/s11042-022-12648-y

[40] Ansari, L., Ji, S.X., Chen, Q. and Cambria, E. (2022) Ensemble Hybrid Learning Methods for Automated Depression Detection. *IEEE Transactions on Computational Social Systems*, **10**, 211-219. https://doi.org/10.1109/TCSS.2022.3154442

[41] Kayalvizhi, S. and Thenmozhi, D. (2022) Data Set Creation and Empirical Analysis for Detecting Signs of Depression from Social Media Postings. arXiv: 2202.03047.

[42] Kabir, M.K., Islam, M., Binte Kabir, A.N., Haque, A. and Rhaman, M.K. (2022) Detection of Depression Severity Using Bengali Social Media Posts on Mental Health: Study Using Natural Language Processing Techniques. *JMIR Formative Research*, **6**, e36118. https://doi.org/10.2196/36118

[43] Baghdadi, N.A., Malki, A., Balaha, H.M., AbdulAzeem, Y., Badawy, M. and Elhosseini, M. (2022) An Optimized Deep Learning Approach for Suicide Detection through Arabic Tweets. *PeerJ Computer Science*, **8**, e1070. https://doi.org/10.7717/peerj-cs.1070

[44] Nalini, N. and Rathi, S. (2022) Depression Prediction in Social Networks Using Whale Optimization Algorithm Based Convolution Neural Networks (WOA-CNN). *International Journal of Early Childhood Special Education*, **31**, 41-52.

[45] Ferreira, R., Trifan, A. and Oliveira, J.L. (2022) Early Risk Detection of Mental Illnesses Using Various Types of Textual Features. Working Notes of CLEF, 5-8.

[46] Kumar, P., Samanta, P., Dutta, S., Chatterjee, M. and Sarkar, D. (2022) Feature Based Depression Detection from Twitter Data Using Machine Learning Techniques. *Journal of Scientific Research*, **66**, 220-228. https://doi.org/10.37398/JSR.2022.660229

[47] Tay, H.E., Lim, M.K. and Chong, C.Y. (2022) Sercnn: Stacked Embedding Recurrent Convolutional Neural Network in Detecting Depression on Twitter. arXiv: 2207.14535.

[48] Lia, R.J., Siddikk, A.B., Muntasir, F., Motiur Rahman, S.S.M. and Jahan, N. (2022) Depression Detection from Social Media Using Twitter's Tweet. In: Baddi, Y., Gahi, Y., Maleh, Y., Alazab, M. and Tawalbeh, L., Eds., *Big Data Intelligence for Smart Applications*, Springer, Cham, 209-226. https://doi.org/10.1007/978-3-030-87954-9_9

[49] Shankdhar, A., Mishra, R. and Shukla, N. (2022) An Application of Deep Learning in Identification of Depression among Twitter Users. In: Khanna, A., Gupta, D., Bhattacharyya, S., Hassanien, A.E., Anand, S. and Jaiswal, A., Eds., *International Conference on Innovative Computing and Communications*, Springer, Singapore, 661-669. https://doi.org/10.1007/978-981-16-3071-2_54

[50] Tong, L., Liu, Z.H., Jiang, Z.H., Zhou, F.X., Chen, L., Lyu, J.L., Zhang, X.R., Zhang, Q.N., Sadka, A., Wang, Y.H., *et al.* (2022) Cost-Sensitive Boosting Pruning Trees for

Depression Detection on Twitter. *IEEE Transactions on Affective Computing*, **14**, 1898-1911. https://doi.org/10.1109/TAFFC.2022.3145634

[51] Chen, Z.Y., Yang, R., Fu, S.Y., Zong, N.S., Liu, H.F. and Huang, M. (2023) Detecting Reddit Users with Depression Using a Hybrid Neural Network. arXiv: 2302.02759.

[52] Muzafar, D., Khan, F.Y. and Qayoom, M. (2023) Machine Learning Algorithms for Depression Detection and Their Comparison. arXiv: 2301.03222.

[53] Vasha, Z.N., Sharma, B., Esha, I.J., Al Nahian, J. and Polin, J.A. (2023) Depression Detection in Social Media Comments Data Using Machine Learning Algorithms. *Bulletin of Electrical Engineering and Informatics*, **12**, 987-996. https://doi.org/10.11591/eei.v12i2.4182

[54] Surekha Reddy, B., Kondaveti, J., Bhavani, V.A. and Aishwarya, P. (2023) Development of a Depression Detection System Using Speech and Text Data. Technical Report, EasyChair.

[55] Noor, F., Iqbal, M.J., Yaqoob, S., Mehmood, S., Jaffar, A. and Ul Haq, I. (2023) Depression Detection in Social Media Using Bagging Classifier. *Journal of Jilin University* (*Engineering and Technology Edition*), **42**, 53-75.

[56] Yang, A.M. (2023) Natural Language Processing of Narrative Writing for Depression Screening in Adolescents. https://doi.org/10.31234/osf.io/5f9nb

[57] Go, A., Bhayani, R. and Huang, L. (2009) Twitter Sentiment Classification Using Distant Supervision. CS224N Project Report, Stanford.

[58] Yates, A., Cohan, A. and Goharian, N. (2017) Depression and Self-Harm Risk Assessment in Online Forums. *Proceedings of the* 2017 *Conference on Empirical Methods in Natural Language Processing*, Copenhagen, September 2017, 2968-2978. https://doi.org/10.18653/v1/D17-1322

[59] Loria, S., *et al.* (2018) Textblob Documentation. Release 0.15.

[60] Mikolov, T., Chen, K., Corrado, G. and Dean, J. (2013) Efficient Estimation of Word Representations in Vector Space. arXiv: 1301.3781.

[61] Pennington, J., Socher, R. and Manning, C.D. (2014) Glove: Global Vectors for Word Representation. *Proceedings of the* 2014 *Conference on Empirical Methods in Natural Language Processing* (*EMNLP*), Doha, October 2014, 1532-1543. https://doi.org/10.3115/v1/D14-1162

[62] Neumann, M.P.M., Iyyer, M., Gardner, M., Clark, C., Lee, K. and Zettlemoyer, L. (2018) Deep Contextualized Word Representations. arXiv: 1802.05365.

[63] Abadi, M., Barham, P., Chen, J., Chen, Z.F., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., *et al.* (2016) TensorFlow: A System for Large-Scale Machine Learning. 12*th USENIX Symposium on Operating Systems Design and Implementation* (*OSDI* 16), Savannah, 2-4 November 2016, 265-283.

[64] Rokach, L. and Maimon, O. (2005) Decision Trees. In: Maimon, O. and Rokach, L., Eds., *Data Mining and Knowledge Discovery Handbook*, Springer, Boston, 165-192. https://doi.org/10.1007/0-387-25465-X_9

[65] Geurts, P., Ernst, D. and Wehenkel, L. (2006) Extremely Randomized Trees. *Machine Learning*, **63**, 3-42. https://doi.org/10.1007/s10994-006-6226-1

[66] Chen, T.Q. and Guestrin, C. (2016) Xgboost: A Scalable Tree Boosting System. *Proceedings of the* 22*nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, 13-17 August 2016, 785-794. https://doi.org/10.1145/2939672.2939785

[67] Ke, G.L., Meng, Q., Finley, T., Wang, T.F., Chen, W., Ma, W.D., Ye, Q.W. and Liu, T.Y. (2017) LightGBM: A Highly Efficient Gradient Boosting Decision Tree. 31*st*

Conference on Neural Information Processing Systems (*NIPS* 2017), Long Beach, 4-9 December 2017, 1-9.

[68] Ramchoun, H., Ghanou, Y., Ettaouil, M. and Janati Idrissi, M.A. (2016) Multilayer Perceptron: Architecture Optimization and Training. *Proceedings of the* 2*nd international Conference on Big Data*, *Cloud and Applications*, Tetouan, 29-30 March 2017, 1-6. https://doi.org/10.1145/3090354.3090427

[69] Nadeau, C. and Bengio, Y. (1999) Inference for the Generalization Error. *Proceedings of the* 12*th International Conference on Neural Information Processing Systems*, Denver, 29 November-4 December 1999, 307-313.