Depression and Anxiety Detection on Social Media

INTRODUCTION

ABSTRACT:

This project dives into the working of Natural Language Processing for Sentiment Analysis applying concepts of Data Sciences and Machine Learning. The project explores in detail the process of sentiment analysis right from tagging and labelling data and classifying sentiments manually to creating a ML Model to automate this prediction.

Depression and Anxiety are alarmingly becoming common in today's modern stressful world. Early prediction, especially using text messages on social media platforms will help people seek help and prevent adverse consequences.

The primary objective of this project is to develop a robust and accurate system that can analyse social media messages to identify potential signs of mental health challenges.

With the prevalence of online communication, monitoring individuals' linguistic patterns on social media can provide valuable insights into their emotional well-being.

The project seeks to harness the power of machine learning and NLP algorithms to create a proactive approach to mental health support by identifying at-risk individuals based on their online expressions.

Through the implementation of this project, we aim to contribute to the larger discourse on mental health awareness and destigmatization by providing users with timely resources and assistance

The introduction of this NLP-based system aligns with the vision of creating a more supportive and empathetic online environment, fostering a sense of community and understanding.

The project acknowledges the ethical considerations inherent in monitoring mental health on social media and prioritises user privacy and consent in its design and implementation.

Through this project, we envision a future where technology acts as a positive force in identifying and supporting individuals experiencing mental health challenges, ultimately contributing to a healthier and more compassionate society.

APPLICATIONS:

The application of Depression and Anxiety Detection on Social Media has several potential benefits across various domains. Here are some key applications:

1. **Early Intervention and Support:**
    - Identify individuals showing early signs of depression or anxiety on social media.
    - Provide timely resources, support, or interventions to help users manage their mental health.
2. **Public Health Surveillance:**
    - Monitor and analyse population-level mental health trends through social media data.
    - Gain insights into the prevalence of mental health issues in different demographics or geographic locations.
3. **Research and Epidemiology:**
    - Contribute to mental health research by providing a large-scale dataset for studying patterns and risk factors associated with depression and anxiety.
    - Enhance our understanding of how social media behaviours correlate with mental health outcomes.
4. **Customised Mental Health Services:**
    - Tailor mental health services and resources based on the specific needs identified through social media data.
    - Personalise interventions to address individual concerns and challenges.
5. **Community Building and Support:**
    - Foster online communities that promote mental health awareness and destigmatization.
    - Encourage supportive interactions among users and create a positive online environment.

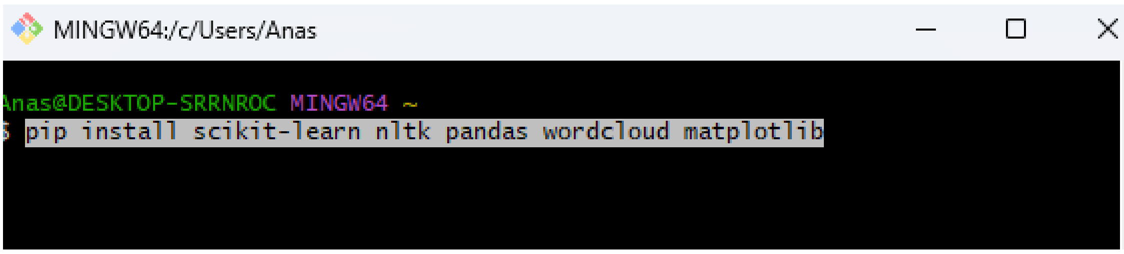
6. **Telehealth Integration:**
    - Facilitate the integration of mental health services into telehealth platforms.
    - Enhance the efficiency of virtual mental health consultations by leveraging social media data for initial assessments.
7. **Education and Awareness Campaigns:**
    - Use insights from social media data to design targeted mental health education and awareness campaigns.
    - Reach specific demographics with information about mental health resources and coping strategies.
8. **Corporate Wellness Programs:**
    - Assist employers in identifying signs of stress, anxiety, or depression among their employees through social media monitoring.
    - Implement workplace wellness initiatives based on the identified needs of the workforce.
9. **Crisis Response and Emergency Services:**
    - Integrate depression and anxiety detection tools into crisis response systems to identify individuals in immediate need of assistance.
    - Improve the effectiveness of emergency mental health services.
10. **Machine Learning Model Improvement:**
    - Continuously refine and improve machine learning models through the feedback loop of real-time social media data.
    - Enhance the accuracy and sensitivity of algorithms for better mental health detection.

It's important to **note** that the **ethical considerations, user privacy, and consent should be central to the implementation of these applications to ensure the responsible use of sensitive mental health** data on social media.

REQUIREMENTS OF ARTIFICIAL INTELLIGENCE:

To **physically implement an AI system for Depression and Anxiety Detection** using NLP on Social Media, you'll need to consider the following requirements:

1. **Python Environment:**
    - Ensure that you have Python installed on your system.
2. **Libraries and Modules:**
    - Install necessary libraries using a package manager such as pip.
    - You need **scikit-learn, NLTK, pandas, PIL, wordcloud,** and **matplotlib** modules.



3. **NLP Tools:**
    - Download NLTK resources for WordNet and stopwords using `nltk.download('wordnet')` **and** `nltk.download('stopwords')`.
4. **Text Preprocessing:**
    - Preprocessing is crucial. Make sure you have functions or methods to perform tasks like text **cleaning, lemmatization**, and **removing stopwords.**
5. **Data Loading:**
    - Prepare your dataset for training and testing. The code snippet suggests using pandas to load and manipulate data.
6. **Feature Extraction:**
    - Utilise TfidfVectorizer or other vectorization techniques to convert textual data into numerical features that machine learning models can process.
7. **Model Selection:**
    - We have to choose appropriate machine learning models for classification. The snippet includes **K-Nearest Neighbors (KNN), Decision Tree**, and **Logistic Regression.** You may experiment with other models based on the nature of your data.

8. **Training and Testing:**
    - Split your dataset into training and testing sets using `train_test_split` from scikit-learn.
9. **Model Evaluation:**
    - Utilise metrics from scikit-learn (e.g., `classification_report`) to evaluate the performance of your models.
10. **Word Cloud Visualization:**
    - Use the provided code for creating word clouds to visualise prominent words in the dataset.
11. **Dependencies:**
    - Ensure that you have all the necessary dependencies installed for the code to run smoothly, such as Jupyter Notebook if you are using it.

Here's my **code snippets** from **Google Colab** for your considerations:

python

```python
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.metrics import classification_report

import nltk
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
import pandas as pd

from PIL import Image
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Download NLTK resources
nltk.download('wordnet')
nltk.download('stopwords')
```

TECHNICAL CHALLENGES:

Creating a Depression and Anxiety Detection system using NLP on Social Media presents several **technical challenges**. Here are **some key challenges to consider**:

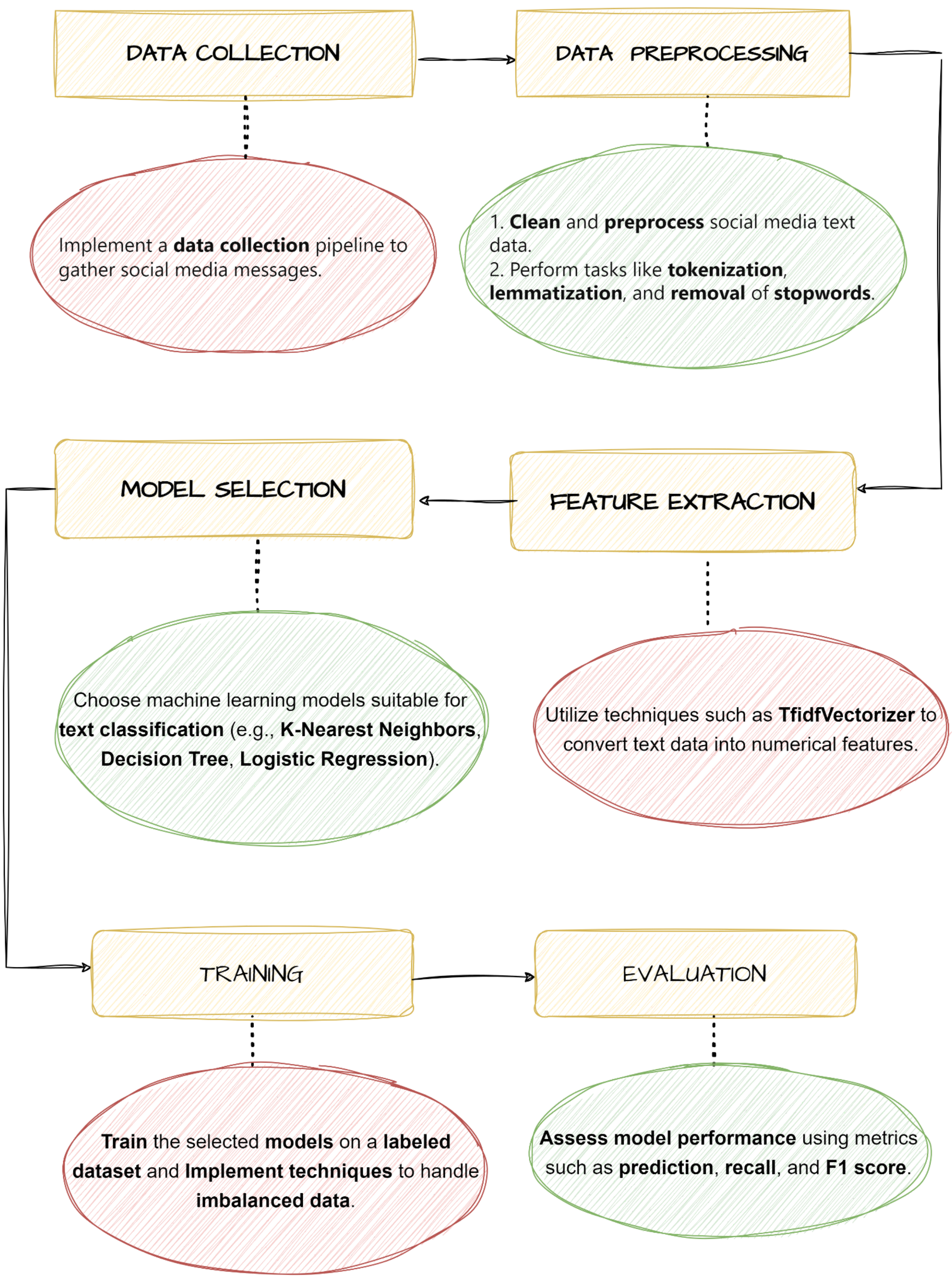
1. **Data Quality and Diversity:**
    - **Challenge:** Obtaining a diverse and representative dataset of social media messages that accurately reflects the nuanced language associated with mental health.
    - **Solution:** Implement data collection strategies that encompass a wide range of demographics, cultures, and mental health expressions.
2. **Labelling and Annotation:**
    - **Challenge:** Accurately labelling social media data with mental health indicators can be subjective and challenging.
    - **Solution:** Establish clear annotation guidelines and use multiple annotators to enhance label accuracy.
3. **Privacy and Ethical Concerns:**
    - **Challenge:** Balancing the need for mental health insights with user privacy and ethical considerations.
    - **Solution:** Implement strict privacy protocols, anonymize data, and obtain explicit consent from users for the analysis of their social media content.
4. **Contextual Understanding:**
    - **Challenge:** Capturing the context and nuances of language to distinguish between expressions of genuine mental health concerns and other uses of related vocabulary.
    - **Solution:** Develop sophisticated NLP models that consider context, temporal aspects, and user history for more accurate detection.
5. **Handling Imbalanced Data:**
    - **Challenge:** Social media data may be imbalanced, with fewer instances of mental health indicators compared to neutral content.
    - **Solution:** Implement techniques such as oversampling, undersampling, or using appropriate class weights to address imbalances during model training.
6. **Real-time Processing:**
    - **Challenge:** Enabling real-time processing of social media messages for timely intervention.
    - **Solution:** Optimise algorithms and system architecture for efficiency, potentially using streaming data processing technologies.

7. **Interpretable Models:**
    - **Challenge:** Ensuring the interpretability of models to understand the features influencing mental health predictions.
    - **Solution:** Choose or design models with explainability features, and prioritise transparency in the decision-making process.
8. **Cross-Cultural Sensitivity:**
    - **Challenge:** Adapting the model to account for cultural variations in expressing mental health concerns.
    - **Solution:** Train models on diverse datasets that include various cultural and linguistic expressions of mental health.
9. **Continuous Model Improvement:**
    - **Challenge:** Ensuring the model remains effective over time as language use evolves.
    - **Solution:** Implement continuous learning mechanisms, regularly update training data, and stay informed about emerging linguistic trends.
10. **Integration with Support Systems:**
    - **Challenge:** Effectively integrating the AI system with existing mental health support systems.
    - **Solution**: Collaborate with mental health professionals to design seamless integration, ensuring the system complements human intervention.

Addressing these challenges requires a multidisciplinary approach involving some expertise in machine learning, natural language processing, mental health, and ethical considerations.

MY WORK:

SYSTEM DIAGRAM:



SENTENCE PRE-PROCESSING:

For **sentence preprocessing** in the context of the Depression and Anxiety Detection project, you would typically perform a series of text cleaning and remove stopwords to prepare the social media messages for analysis.Here's an example of sentence preprocessing steps using Python and the NLTK library:

1. Tokenization :

- In this step, the text is split into smaller units. We can use either sentence tokenization or word tokenization based on our problem statement.

- **Tokenization** is done using the module gensim.utils.simple_preprocess which tokenizes the sentence, converts to lowercase and removes punctuations.

2. Stop Word Removal :

- **Stopwords** are the commonly used words and are removed from the text as they do not add any value to the analysis. These words carry less or no meaning.

- **NLTK library consists of a** list of words that are considered stopwords for the English language. Some of them are : [i, me, my, myself, we, our, ours, ourselves, you, you're, you've, you'll, you'd, your, yours, yourself, yourselves, he, most, other, some, such, no, nor, not, only, own, same, so, then, too, very, s, t, can, will, just, don, don't, should, should've, now, d, ll, m, o, re, ve, y, ain, aren't, could, couldn't, didn't, didn't].

```python
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')

def remove_stopwords(text):
    stop_words = set(stopwords.words('english'))
    tokens = nltk.word_tokenize(text)
    filtered_text = [word for word in tokens if word.lower() not in stop_words]
    return ' '.join(filtered_text)

# Example usage
sample_text = "This is an example sentence with some stopwords that need to be removed."
text_without_stopwords = remove_stopwords(sample_text)

print("Original Text:", sample_text)
print("Text without Stopwords:", text_without_stopwords)
```

3. Singularize :

- **Singularize** is used to **convert** a plural word to its singularize form.

- **Singularize module** is used from pattern library to convert the words into its singularize form. However multiple modules are also available to singularize like textblob from Nltk, etc.

```
import nltk
from nltk.stem import WordNetLemmatizer

# Download popular datasets including WordNet
nltk.download('popular')

def singularize_word(word):
    lemmatizer = WordNetLemmatizer()
    return lemmatizer.lemmatize(word, pos='n')
```

4. Lemmatization :

- It stems from the word but makes sure that it does not lose its meaning. Lemmatization has a pre-defined dictionary that stores the context of words and checks the word in the dictionary while diminishing.

- Lemma module from the pattern library is used for lemmatization . However, multiple modules are available for lemmatization like gensim.utils modules, etc.

```
import nltk
from nltk.stem import WordNetLemmatizer

nltk.download('wordnet')

def lemmatize_text(text):
    lemmatizer = WordNetLemmatizer()
    tokens = nltk.word_tokenize(text)
    lemmatized_tokens = [lemmatizer.lemmatize(word) for word in tokens]
    return ' '.join(lemmatized_tokens)
```

<u>FEATURE EXTRACTION:</u>

In Natural Language Processing, **Feature Extraction** is one of the trivial steps to be followed for a better understanding of the context of what we are dealing with. After the initial text is cleaned and normalised, we need to transform it into their features to be used for modelling. We use some particular method to assign weights to particular words within our document before modelling them. We go for numerical representation for individual words as it's easy for the computer to process numbers.

Vectorization is the process of converting textual data into numerical vectors that can be used as input for machine learning models or other data analysis tasks. It is a crucial step in natural language processing (NLP) and text analysis. There are several methods for vectorizing text data, and one of the commonly used methods is the Term Frequency-Inverse Document Frequency (TF-IDF) vectorization. Here's an overview of the vectorization process using TF-IDF:

Here we have used **Word2Vec,** one of the methods of Vectorization.

Word2Vec is widely used in most of the NLP models. It transforms the word into vectors. Word2vec is a two-layer net that processes text with words. The input is in the text corpus and the output is a set of vectors: feature vectors represent the words on that corpus. While Word2vec is not a deep neural network, it converts text into an unambiguous form of computation for deep neural networks. The purpose and benefit of Word2vec is to collect vectors of the same words together in vector space. That is, it finds mathematical similarities. Word2vec creates vectors that are distributed by numerical presentations of word elements, features such as individual word context. It does so without human intervention.

Word2Vec can capture the contextual meaning of words very well. There are two flavours. In one of the methods, we are given the neighbouring words called the **continuous bag of words (CBoW)**, and in which we are given the middle word called skip-gram and we predict the neighbouring words. Once we get a pre-trained set of weights we can save it and this can be used later for word vectorization without the need for transformation again.

Here's an overview of the **vectorization** process **using TF-IDF**:

1. **Tokenization:** The first step is to tokenize the text, which means splitting it into individual words or tokens. Tokenization can also involve handling punctuation, converting text to lowercase, and other text preprocessing steps.

2. **Counting Term Frequency (TF):** TF measures how frequently each term (word) appears in a document. For each document in your corpus, you create a vector where each element represents the frequency of a specific term within that document.

3. **Inverse Document Frequency (IDF):** IDF measures the importance of each term in the entire corpus. It helps to identify words that are unique or specific to certain documents. Terms that appear frequently across many documents have lower IDF scores.

4. **TF-IDF Calculation:** The TF-IDF score for a term in a document is calculated by multiplying the TF and IDF values. It highlights terms that are both frequent within a document and unique to that document.

5. **Vectorization:** Once you have the TF-IDF scores for all the terms in a document, you can represent each document as a vector. The vector typically contains TF-IDF scores for all the terms in your vocabulary. If you have a large vocabulary, you may use a subset of the most important terms to reduce dimensionality.

6. **Machine Learning or Analysis:** The resulting TF-IDF vectors can be used as input to machine learning models (e.g., classification, clustering) or for various text analysis tasks such as information retrieval or sentiment analysis.

The **TF-IDF vectorization process** is just one of many techniques for text vectorization. Other methods, such as **Word2Vec**, **Doc2Vec**, and **Bag of Words** (BoW), create vector representations using different strategies. The choice of vectorization method depends on the specific NLP task and the characteristics of your text data.

CLASSIFIER USED:

K-Nearest Neighbors, Decision Trees and **Logistic Regression** are diverse algorithms used which depends on the characteristics of the data shows considerations such as interpretability and computational efficiency:

1. ## K-Nearest Neighbors (KNN):

   K-Nearest Neighbors is a supervised machine learning algorithm used for both classification and regression tasks. It is a non-parametric and lazy algorithm, meaning it doesn't make assumptions about the underlying data distribution during training and postpones the learning process until predictions are needed.

   Working Mechanism:

   1.Training: In the training phase, the algorithm stores the entire training dataset.

   2.Prediction (Classification): To make a prediction for a new data point, KNN identifies the K-nearest neighbours in the training dataset based on a distance metric (commonly Euclidean distance). The majority class among these neighbours is assigned to the new data point.

   3.Prediction (Regression): For regression tasks, the algorithm calculates the average of the target values of the K-nearest neighbours.

   Strengths:

   - Simplicity and ease of implementation.
   - No assumptions about the data distribution.

   Weaknesses:

   - Can be computationally expensive, especially with large datasets.
   - Sensitive to irrelevant or redundant features.

2. Decision Tree:

A Decision Tree is a tree-like model used for both classification and regression tasks. It partitions the data into subsets based on features at each node and makes decisions by traversing the tree from the root to a leaf node.

Working Mechanism:

1. Node Splitting: The algorithm selects the best feature to split the data at each node based on criteria such as Gini impurity (for classification) or mean squared error (for regression).

2. Tree Growing: The process is repeated recursively until a stopping criterion is met, such as a maximum depth or a minimum number of samples per leaf.

3. Prediction: To make a prediction for a new data point, it traverses the tree from the root to a leaf, and the target value or class of the leaf node is assigned.

Strengths:

- Intuitive and easy to understand.
- Handles both numerical and categorical data.

Weaknesses:

- Prone to overfitting, especially with deep trees.
- Sensitive to small variations in the data.

3. Logistic Regression:

Logistic Regression is a binary classification algorithm used to model the probability of a binary outcome (0 or 1). Despite its name, it's a classification algorithm rather than a regression algorithm.

Working Mechanism:

1.Logistic Function: Logistic Regression uses the logistic function (sigmoid) to transform a linear combination of features into a probability between 0 and 1.

2.Decision Boundary: Based on the calculated probabilities, a decision boundary is set. If the probability is above a certain threshold (usually 0.5), the outcome is predicted as class 1; otherwise, it's predicted as class 0.

3.Training: The algorithm estimates the coefficients (weights) of the linear combination during the training phase using methods like maximum likelihood estimation.

Strengths:

- Simple and interpretable.
- Efficient for binary classification.

Weaknesses:

- Assumes a linear relationship between features and the log-odds.
- Limited to binary classification (extensions exist for multi-class problems).

EXPERIMENT

Designing a **database** for Depression and Anxiety Detection on Social Media involves several considerations. Below is a simplified example of how you might structure your database with tables.

User Table

user_id	username	email	other_fields
1	user1	user1@example.com	...
2	user2	user2@example.com	...
...	...	...	...

Social Media Post Table

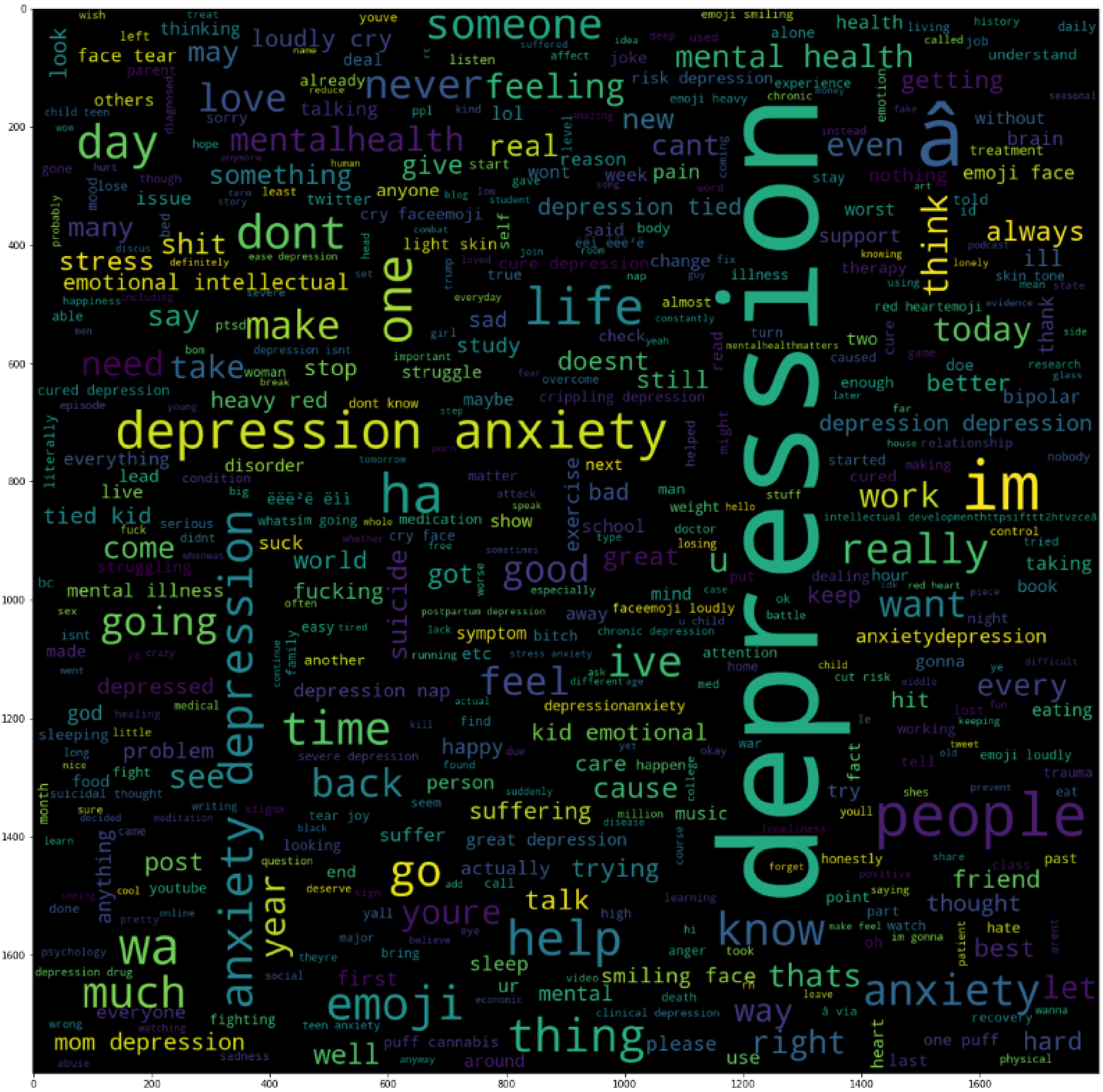
post_id	user_id	post_text	timestamp
101	1	Feeling really down today...	2023-01-15 12:30
102	2	Can't shake off this anxiety...	2023-01-16 15:45
...	...	...	...

The **user table** stores information about the users who are part of the social media platform and The **social media post table** contains posts made by users on the social media platform.

We should consider the dataset like above and then **start cleaning data** => Removing Stop Words + Lemmatization + Remove [non-word characters, single characters, spaces].

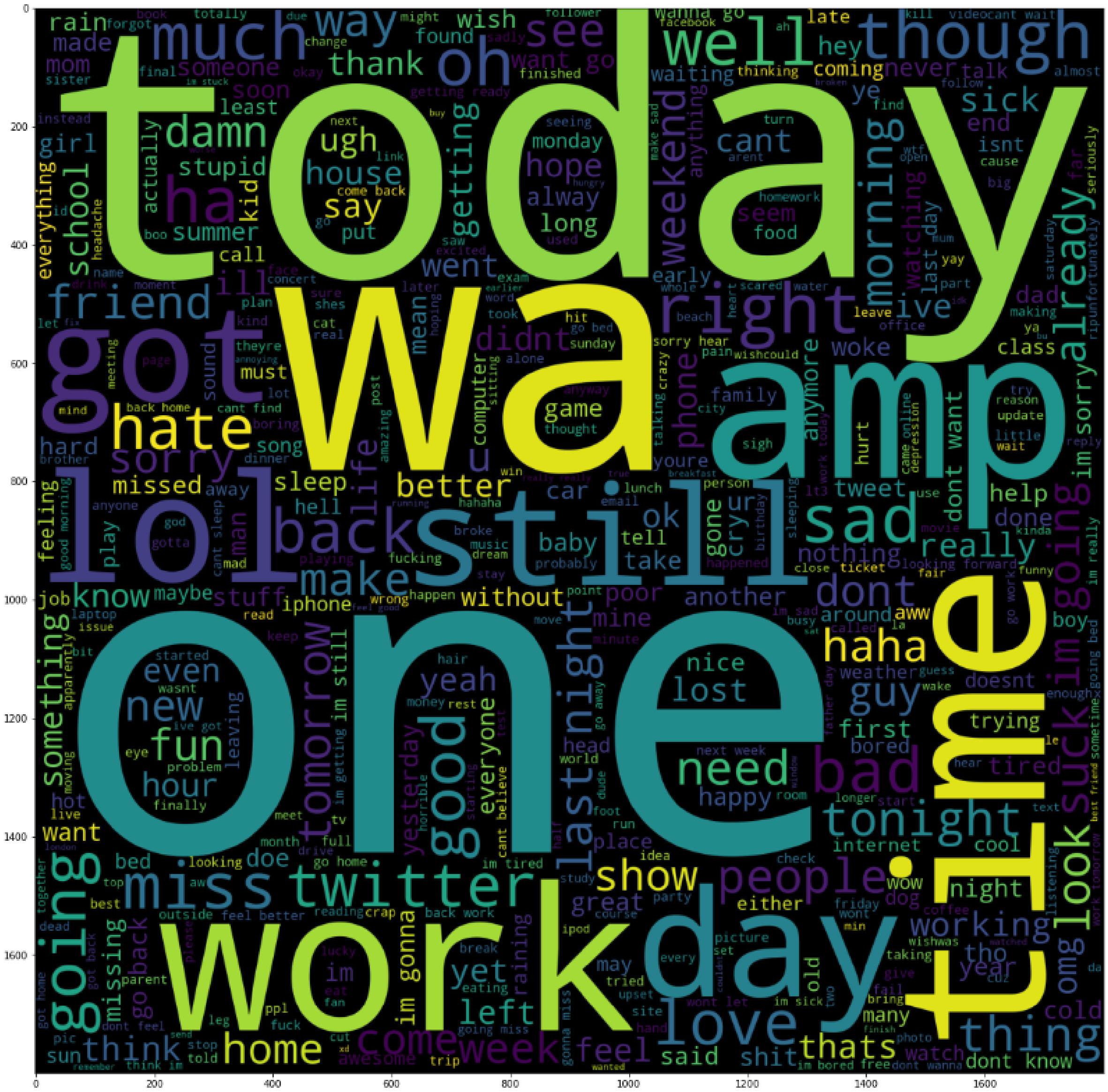
After cleaning the data **we will analyse the words using sentiment analysis** and we get as follows:

Word Cloud on Depression Sentiment consists of **Positive + Negative Tweets.**



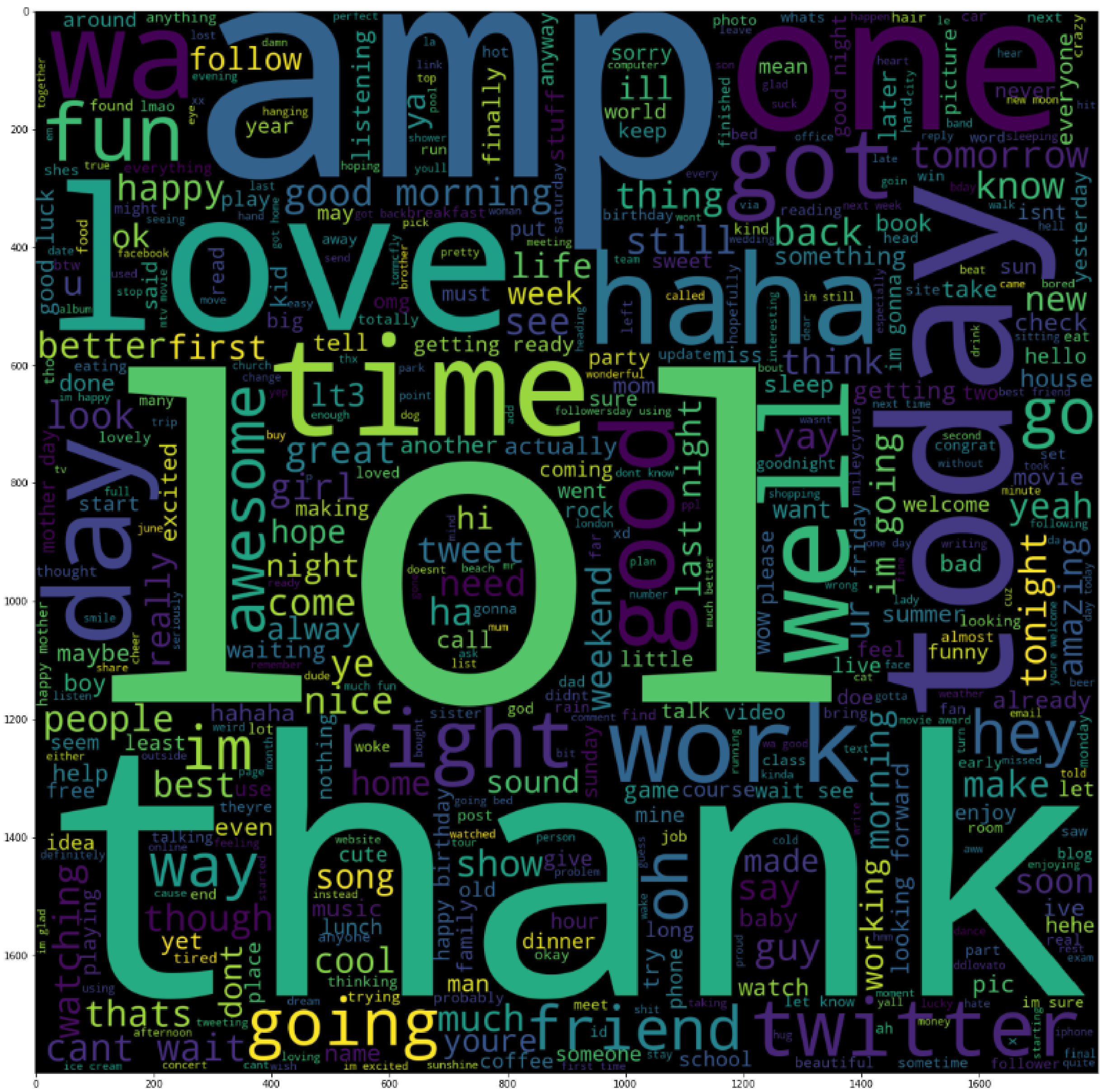
A **Word Cloud** generated from **Depression-related sentiment** analysis encompasses a combination of both positive and negative sentiments expressed in tweets. By merging these sentiments into a single visualisation, the Word Cloud provides a comprehensive overview of the diverse emotions associated with the topic of depression on social media.

Word Cloud on Depressive + Negative Tweets:



A **Word Cloud** generated specifically from **depressive and negative tweets** provides a focus on mental health discourse on social media. By isolating and visualising words associated with negativity and depressive sentiments, the Word Cloud highlights the prevalent themes and expressions related to distress, sadness, and emotional struggles.

Word Cloud on Positive Tweets:



A **Word Cloud** crafted from **positive tweets** provides a visually uplifting representation of the affirming and hopeful sentiments expressed within the reality of mental health discussions on social media. With the positive encouragement and resilience, the Word Cloud offers a powerful snapshot of the constructive aspects of these online conversations.

Thus, we **further** vectorize using Word2Vec **method of Vectorization** after semantic analyzation and used the classifiers (K-Nearest Neighbors (KNN)**,** Decision Tree**,** and Logistic Regression).

After using the classifier's as a decision making we will finalise the testing model and analyse the result as follow:

RESULT:

Sentiment Analysis Result Table

post_id	sentiment_score	sentiment_category
101	-0.75	Negative
102	-0.60	Negative
...	...	...

It shows the sentiment analysis of the data into negative and positive tweets accordingly. After analysing all the data the code comes to its **predicted calculations** and the observations are:

```
              precision    recall  f1-score   support

           0       0.83      0.81      0.82    161744
           1       0.95      0.83      0.89       430
           4       0.81      0.83      0.82    159889

    accuracy                           0.82    322063
   macro avg       0.86      0.82      0.84    322063
weighted avg       0.82      0.82      0.82    322063

done
0.8186814381037251
done
```

Thus, the experiment came to an end and hence predicting the score with more than 80% accuracy is more than good.

Acknowledgement

We extend our deepest gratitude to **Saiyed Umer Sir** who taught us the development and realisation of the Depression Detection on Social Media project. This project represents a collaborative effort and would not have been possible without the dedication and support of many individuals.

We express our sincere thanks to the users who have shared their insights and experiences on social media, making valuable contributions to the dataset that underlies the project. Your willingness to engage in open conversations about mental health has been instrumental in advancing our understanding of the complexities surrounding depression.

Again a special appreciation to my teacher, Umer Sir for teaching me from the very beginning i.e from the implementation of the project, from data collection and preprocessing to the design and optimization of classifiers. Your technical expertise and commitment to the project have been indispensable in shaping its success.

Lastly, we appreciate the users who engage with and benefit from the Depression Detection on Social Media project in future. Your feedback and participation contribute to its ongoing refinement and improvement.

Thank you to everyone involved for your collaborative spirit and commitment to making a positive impact on mental health through technology.