A Project Report On

PREDICTIVE ANALYSIS OF DEPRESSION USING NATURAL LANGUAGE PROCESSING

Mini project submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY
IN
INFORMATION TECHNOLOGY
(2021-2025)
BY

A. VAIBHAV SHANKAR	**21241A1270**
CH. ABHINAV	**21241A1279**
G. AKASH REDDY	**21241A1288**

Under the Esteemed Guidance
of
MR. K. SANDEEP
Assistant Professor

DEPARTMENT OF INFORMATION TECHNOLOGY
GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)
HYDERABAD

2023-24

CERTIFICATE

This is to certify that it is a bonafide record of Mini Project work entitled **"PREDICTIVE ANALYSIS OF DEPRESSION USING NATURAL LANGUAGE PROCESSING"** done by **A. VAIBHAV SHANKAR (21241A1270), CH. ABHINAV (21241A1279), G. AKASH REDDY (21241A1288)** of **B. Tech** in the Department of Information of Technology, **Gokaraju Rangaraju Institute of Engineering and Technology** during the period 2021-2025 in the partial fulfillment of the requirements for the award of degree of **BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY** from GRIET, Hyderabad.

Mr. K. SANDEEP
Assistant Professor
(Internal Guide)

Dr. Y J Nagendra Kumar
Head of the Department

(Project External)

ACKNOWLEDGEMENT

We take the immense pleasure in expressing gratitude to our Internal guide**, Mr. K. SANDEEP, Assistant Professor, Dept of IT**, GRIET. We express our sincere thanks for his encouragement, suggestions and support, which provided the impetus and paved the way for the successful completion of the project work.

We wish to express our gratitude to **Dr. Y J Nagendra Kumar,** HOD IT, our Project Coordinators **Dr. R.V.S.S.S.Nagini** and **Mr. P.K. Abhilash** for their constant support during the project.

We express our sincere thanks to **Dr. Jandhyala N Murthy,** Director, GRIET**,** and **Dr. J. Praveen,** Principal, GRIET**,** for providing us the conductive environment for carrying through our academic schedules and project with ease.

We also take this opportunity to convey our sincere thanks to the teaching and non-teaching staff of GRIET College, Hyderabad.

Email: vaibhavshankara@gmail.com

Contact: 7036894236

Email: chitikesiabhinav510@gmail.com

Contact: 7330730941

Email: gopidiakashreddy@gmail.com

Contact: 9110548579

DECLARATION

This is to certify that the mini project entitled "**PREDICTIVE ANALYSIS OF DEPRESSION USING NATURAL LANGUAGE PROCESSING"** is a bonafide work done by us in partial fulfilment of the requirements for the award of the degree **BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY** from Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad.

We also declare that this project is a result of our own effort and has not been copied or imitated from any source. Citations from any websites, books and paper publications are mentioned in the Bibliography.

This work was not submitted earlier at any other University or Institute for the award of any degree.

A. VAIBHAV SHANKAR	**21241A1270**
CH. ABHINAV	**21241A1279**
G. AKASH REDDY	**21241A1288**

TABLE OF CONTENTS

11 LIST OF DIAGRAMS

ABSTRACT

Depression, a pervasive mental health disorder, poses significant challenges to individuals' well-being and society's health resources. Early detection and intervention are crucial to mitigating its adverse effects. Natural Language Processing (NLP) emerges as a promising tool for predictive analysis in mental health. This study aims to explore the application of NLP techniques for predicting depression based on textual data. The research leverages datasets comprising diverse textual sources, such as social media posts, electronic health records, and online forums, to capture nuanced linguistic patterns indicative of depressive symptoms.

By employing advanced NLP algorithms, including sentiment analysis, topic modelling, and linguistic feature extraction, predictive models are developed to discern subtle linguistic cues associated with depression. Key themes identified in the textual data, such as negative affect, social isolation, and cognitive distortions, serve as pivotal indicators for depression prediction. Through the integration of machine learning methodologies, predictive models achieve notable accuracy in identifying individuals at risk of depression, facilitating timely interventions and personalized treatment strategies.

This interdisciplinary approach not only enhances our understanding of the linguistic markers of depression but also provides actionable insights for healthcare professionals and policymakers to implement targeted interventions and allocate resources effectively. Ultimately, the utilization of NLP in depression prediction signifies a transformative paradigm in mental health care, fostering proactive measures for promoting psychological well-being and societal resilience.

Key Words: Depression, Intervention, NLP, Sentiment Analysis, Interdisciplinary Approach

Domain: Natural Language Processing

1. INTRODUCTION

1.1 Introduction to Project

Mental health challenges, especially depression, are widespread and can have serious consequences for individuals and society. Timely identification and intervention play a crucial role in lessening the negative impacts of depression. While traditional diagnostic approaches are effective, they often rely on self-reported information and clinical assessments, which may not always capture the intricate and changing aspects of an individual's mental well-being. The rise of digital communication has provided a vast amount of textual data from various online sources like social media, blogs, and forums, offering a valuable tool for examining psychological health.

NLP:

Natural Language Processing (NLP) is a multidisciplinary field encompassing linguistics, computer science, and artificial intelligence. It offers effective tools for analyzing and deciphering human language. Through the utilization of NLP methods, researchers can pinpoint linguistic patterns, emotions, and sentiments that may indicate states of depression. These methodologies enable the efficient processing of vast amounts of text data, facilitating continuous and real-time monitoring of mental well-being.

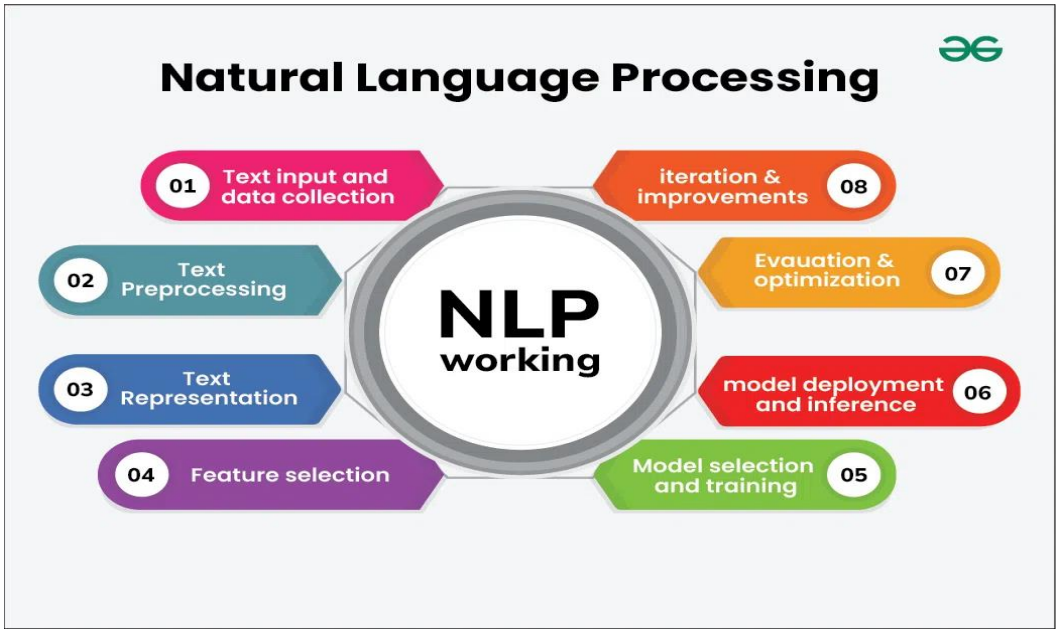


FIG 1: NLP WORKING

The use of natural language processing (NLP) for predicting depression has a wide array of potential benefits. These include offering assistance in clinical diagnosis and creating customized treatment plans, as well as facilitating early intervention via digital mental health platforms. Nonetheless, this strategy comes with its own set of obstacles as privacy issues, the necessity for extensive labeled datasets, and the challenge of accurately understanding human emotions conveyedthrough text.

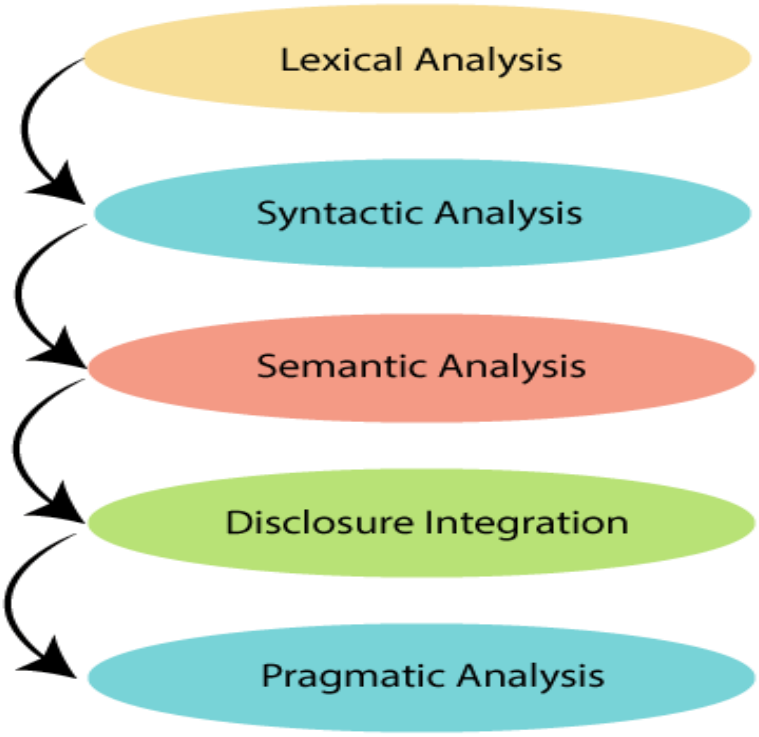


FIG 1.2: PHASES OF NLP

The fusion of NLP with mental health diagnosis is an exciting frontier in the quest to comprehend and tackle depression. Leveraging language data and machine learning enables us to progress towards proactive and precise mental healthcare.

1.2 Existing System

Most of the predictions are done through CNN while CNNs excel at learning hierarchical representations from raw data, they may not capture the full spectrum of relevant features for depression prediction. Important psychosocial factors, such as social interactions, life events, or personality traits, may not be adequately represented in the data or learned by the CNN architecture.

1.3 Proposed System

NLP allows access to rich textual data, such as social media posts, electronic health records, online forums, and clinical notes. This text often contains valuable information about individuals' thoughts, feelings, behavior, and experiences related to depression, providing a wealth of data for analysis.

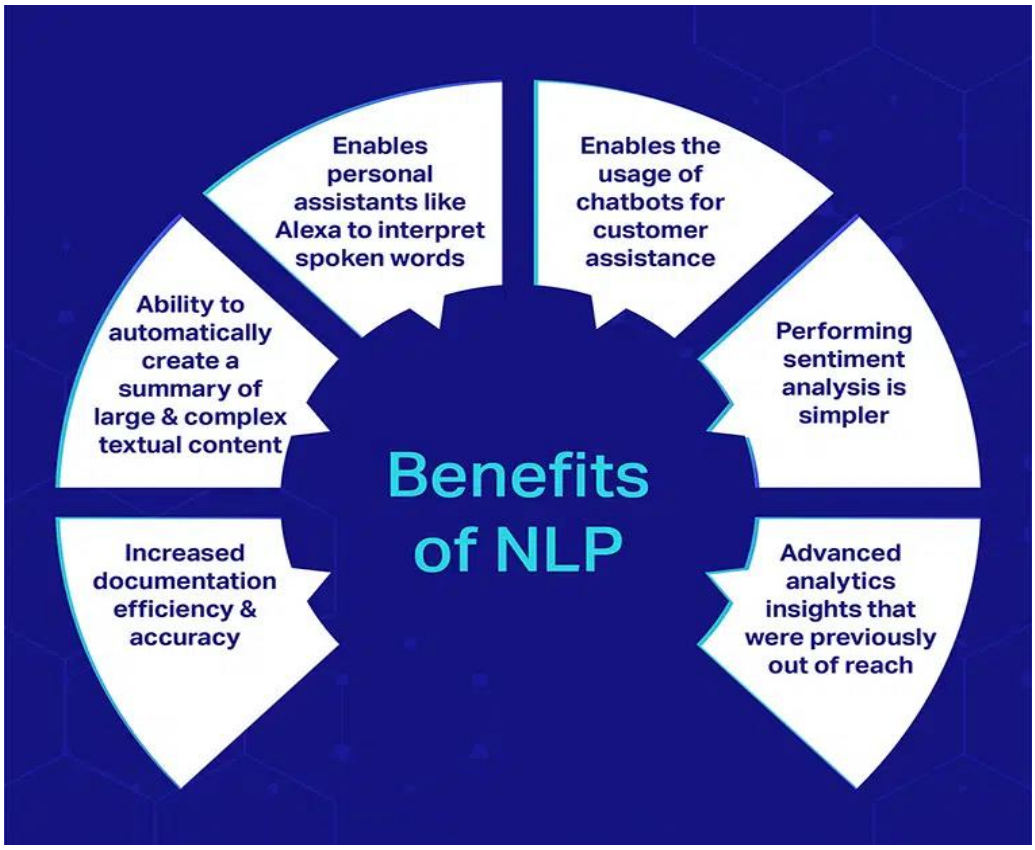


FIG 1.3: BENEFITS OF NLP

2. REQUIREMENT ENGINEERING

2.1 Hardware Requirements

Laptop or PC

Version: Windows 7 or higher

Processor: I3 processor system or higher

Memory: 4 GB RAM or higher

Input devices: Keyboard, Mouse

2.2 Software Requirements

Jupyter Notebook/Google Colab

Python

Python Libraries:

1.pandas

2.numpy

3.nltk

4.beautifulsoup

5.scikit-learn

3. LITERATURE SURVEY

Depression, a prevalent mental health issue with significant personal and societal effects, requires early detection for effective intervention. This thesis aims to develop a reliable system to identify depression from social media text using natural language processing (NLP) and machine learning (ML). Techniques like tokenization, stemming, N-gram, CountVectorizer, and TF-IDF transform raw text into meaningful representations. Machine learning algorithms—SGD, Naive Bayes, Decision Tree, Random Forest, SVM, KNN, MLP—learn patterns from annotated data to distinguish depressive posts. Evaluation metrics such as accuracy, precision, recall, and F1-score demonstrate the system's high performance in detecting depression from social media content.

Web is growing apace in mining concerns using depression detection method form sources such as news, blogs and which also includes product reviews considering the earlier cases the responses given are quite difficult to determine. A technique using Natural Language Processing and Optical Character Recognition. Our experiments are to include that an entity scores that are determined using are statistically different from any other approaches and the quality of the approach is completely independent from any other approaches and scores can be determined reports can be provided for health subscription. Many researchers have demonstrated that by user generated content in a context is a method to determining people's mental health levels. The research is to find out the SNS by user post, which helps in classifying the mental health levels of the user.

The project discusses the prevalence of depression and its impact on daily life, highlighting the limitations of traditional diagnostic methods. To address this, an automated conversational platform is introduced, utilizing Natural Language Processing (NLP) techniques to categorize sentiments into 'happy,' 'neutral,' 'depressive,' and 'suicidal' states. For suicidal states, the platform triggers a call to a prevention helpline, while providing supportive conversations for other states. Integrating Google Home Mini, Google Dialogflow ML, and Twilio API, the system achieves a 76% classification accuracy, surpassing SVM. This pilot study demonstrates promising effectiveness in identifying mental states for timely intervention and support.

Depression is a common mental illness impacting performance and leading to suicidal thoughts. Traditional methods help identify depression, but this work uses machine learning

(ML) and natural language processing (NLP) to predict depressive posts on Reddit. Reddit is ideal due to its timely idea exchange, emotional expression, and medical terminology use. We analyzed comments from r/depression and r/SuicideWatch for insights into suicidal behavior. Using ML algorithms like Naïve Bayes and SVM, we tracked at-risk individuals. Results showed Logistic Regression with 77.29% accuracy and 0.77 F1-score, Naïve Bayes with 74.35% accuracy, SVM with 77.12% accuracy, and Random Forest with 77.30% accuracy.

Mental illness is highly prevalent nowadays, constituting a major cause of distress in people's life with impact on society's health and well-being. Mental illness is a complex multi-factorial disease associated with individual risk factors and a variety of socioeconomic, clinical associations. In order to capture these complex associations expressed in a wide variety of textual data, including social media posts, interviews, and clinical notes, natural language processing (NLP) methods demonstrate promising improvements to empower proactive mental healthcare and assist early diagnosis. We provide a narrative review of mental illness detection using NLP in the past decade, to understand methods, trends, challenges and future directions. A total of 399 studies from 10,467 records were included. The review reveals that there is an upward trend in mental illness detection NLP research. Deep learning methods receive more attention and perform better than traditional machine learning methods. We also provide some recommendations for future studies, including the development of novel detection methods, deep learning paradigms and interpretable models.

4. TECHNOLOGY

4.1 ABOUT PYTHON

Python's ecosystem has transformed significantly over time, enhancing its capabilities for statistical analysis. It effectively balances scalability and organization. Python prioritizes efficiency and readability, featuring a design focused on code readability with a beginner-friendly syntax that enables concise code expression through indentation. Notably, this high-level language boasts dynamic system functions, alongside an automated memory management system.

4.2 APPLICATIONS OF PYTHON

Python is utilized across various fields of applications, establishing a strong foothold in emerging sectors. Recognized as the swiftest-growing programming language, Python has the versatility to develop applications of any nature.

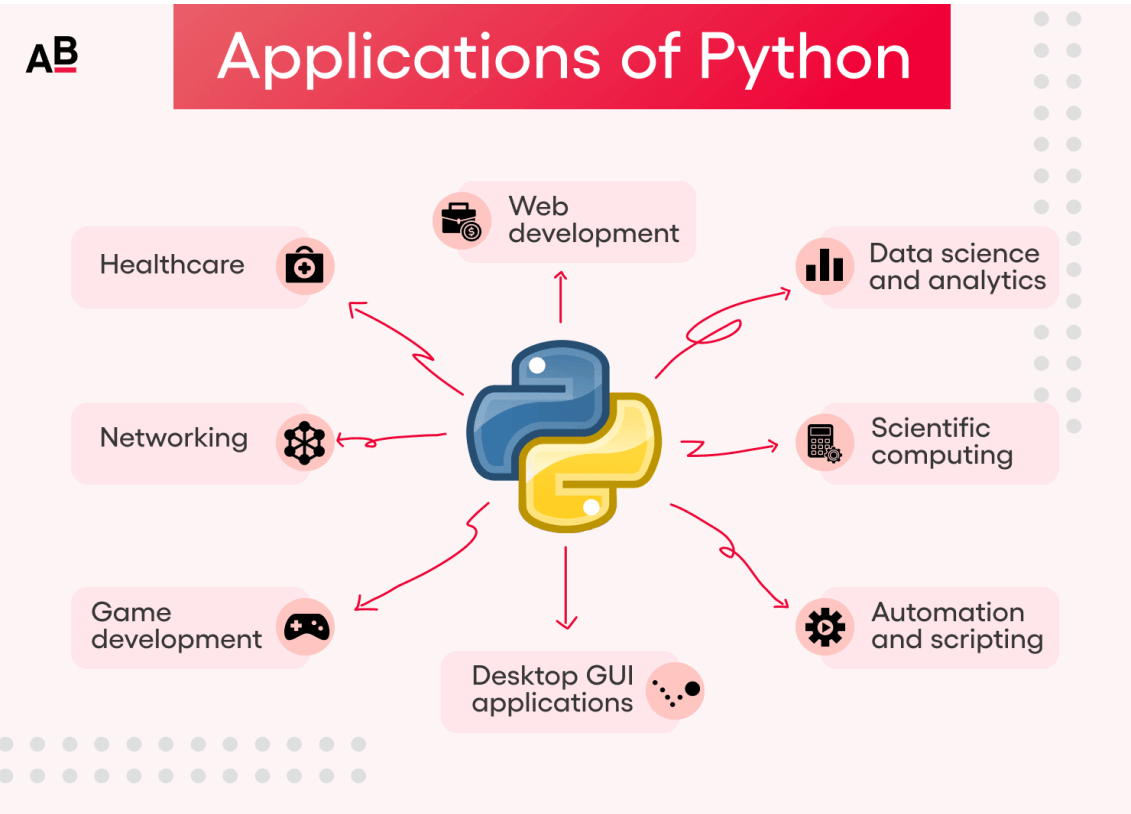


FIG 2: PYTHON APPLICATIONS

It is used in various fields:

- Web Applications. We can use Python to develop web applications. ...
- Desktop GUI Applications. ...
- Console-based Application. ...
- Software Development. ...
- Scientific and Numeric. ...
- Business Applications. ...
- Audio or Video-based Applications. ...

4.3 PYTHON IS WIDELY USED IN DATA SCIENCE

Python data science is utilized in an adaptable and open-source programming language. It offers functionality that pertains to mathematical and scientific tasks. Its popularity stems from its straightforward syntax and vast libraries, significantly reducing coding time. The key libraries in Python for Data Science include:

4.3.1 PANDAS

A Python library is utilized for the analysis, cleaning, exploration, and manipulation of statistics within datasets. Typically, datasets contain both valuable and extraneous data. Pandas enables this data to be easily read and made relevant.

4.3.2 NUMPY

A Python library is utilized for the purpose of reading, cleaning, exploring, and manipulating statistics. Typically, datasets contain both valuable and unnecessary information. Pandas assists in making them easily understandable and relevant.

4.3.3 MATPLOTLIB

A library in Python is employed for creating graphs using NumPy arrays. It enables the plotting of various types of graphs, ranging from basic plots to bar graphs, histograms, scatter plots, and many others.

4.3.4 SCIKIT-LEARN

In Python, Scikitlearn tools are employed for device learning, statistical modeling, including classification, regression, and clustering.

4.3.5 CSV

The file format known as CSV stores tabular data similar to a spreadsheet or database. Each entry contains one or more fields separated by commas. To manipulate CSV files, the csv built-in module is utilized.

4.4 Dataset Description

The dataset has been taken from kaggle to train our model. These datasets are put together into two categories of two different outputs 0 and 1. The dataset consists of 1000 unique texts producing output 0 and 1000 unique texts producing output 1.

Input size: 13.03MB

Why kaggle dataset…?

1. The dataset can be trained easily as it is small in size.
2. Data is properly labelled.
3. The dataset is available for free.

4.5 NLP

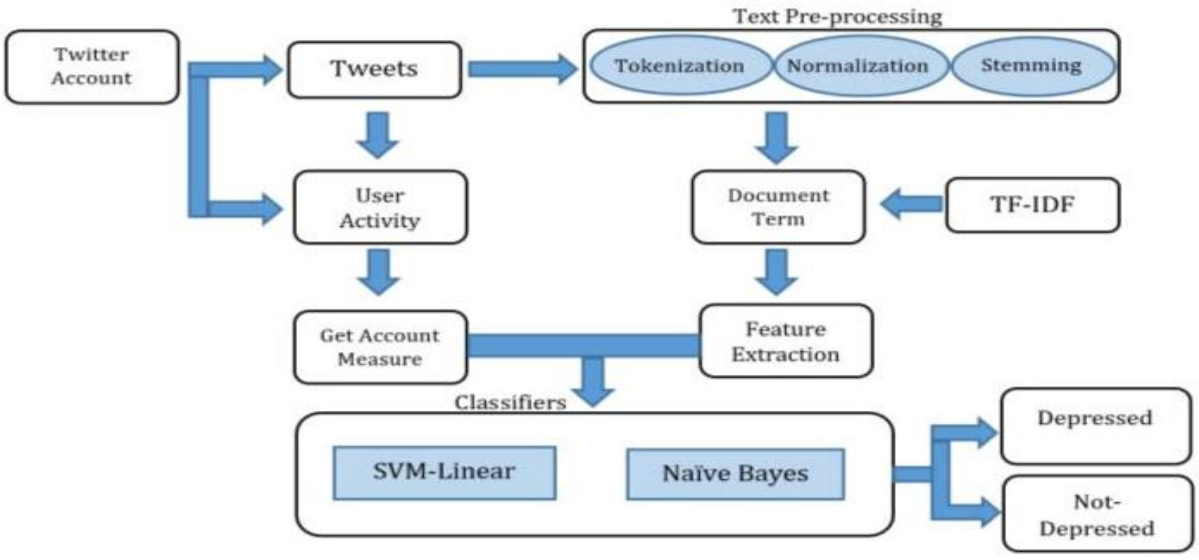


FIG 3: NLP PROCESS

5.DESIGN REQUIREMENT ENGINEERING

Concept of uml:

The goal of these diagrams, created using UML, is to visually showcase the system alongside its main components, functions, actions, elements, or education, with the purpose of enhancing comprehension, control, preservation, or organization of data pertaining to the system.

5.1 UML DIAGRAMS:

The Unified Modeling Language (UML) is utilized to construct models for a variety of purposes. Its primary aim is to offer a standardized approach to visually representing a system's structure, similar to how blueprints are utilized in different branches of engineering. In scenarios involving intricate applications, effective communication between multiple teams is crucial. Since business professionals might not comprehend coding, UML steps in to facilitate conveying vital system requirements, functionalities, and procedures to individuals who aren't programmers. By illustrating processes, user interactions, and the system's static framework, teams can ultimately save time.UML is linked with object-oriented design and analysis. UML makes the use of elements and forms associations between them to form diagrams. Diagrams in UML can be broadly classified as:

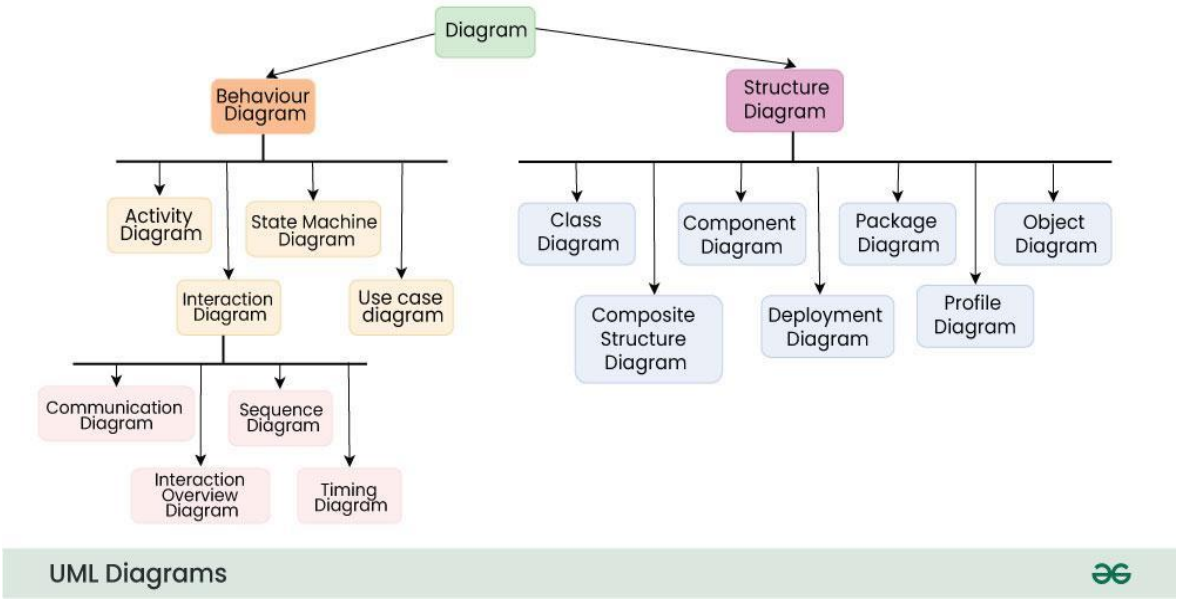


FIG 5.1: CONCEPTS OF UML

5.2 Use case Diagram:

A use case diagram represents the functionalities provided by the system concerning the involved participants, their objectives, and any interdependencies among these scenarios.

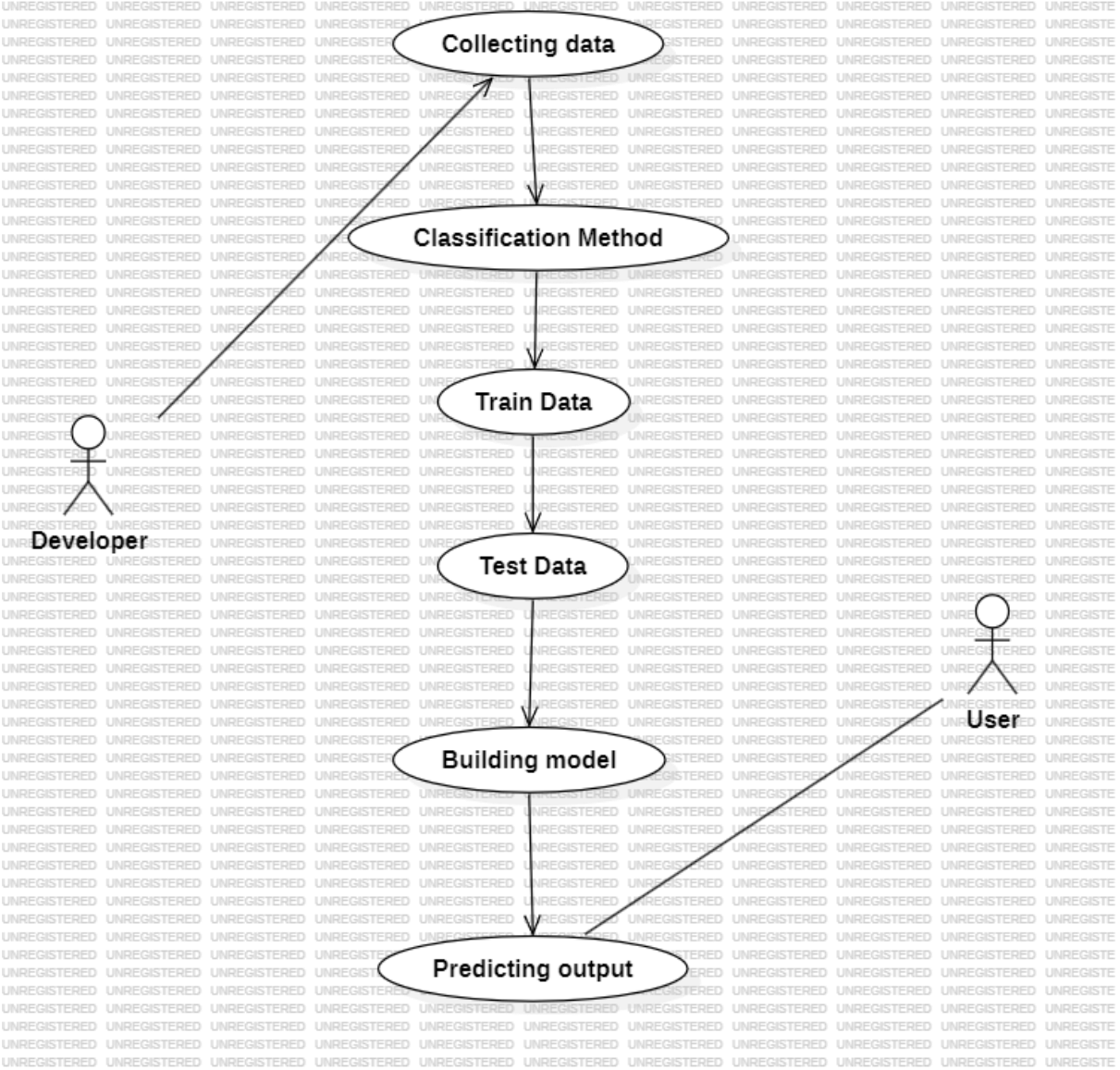


FIG 5.2: USE CASE DIAGRAM

5.3 Class Diagram:

A class diagram is a kind of static structural diagram that illustrates the structure of a system by showing the connections among the system's classes, attributes, operations, and relationships.

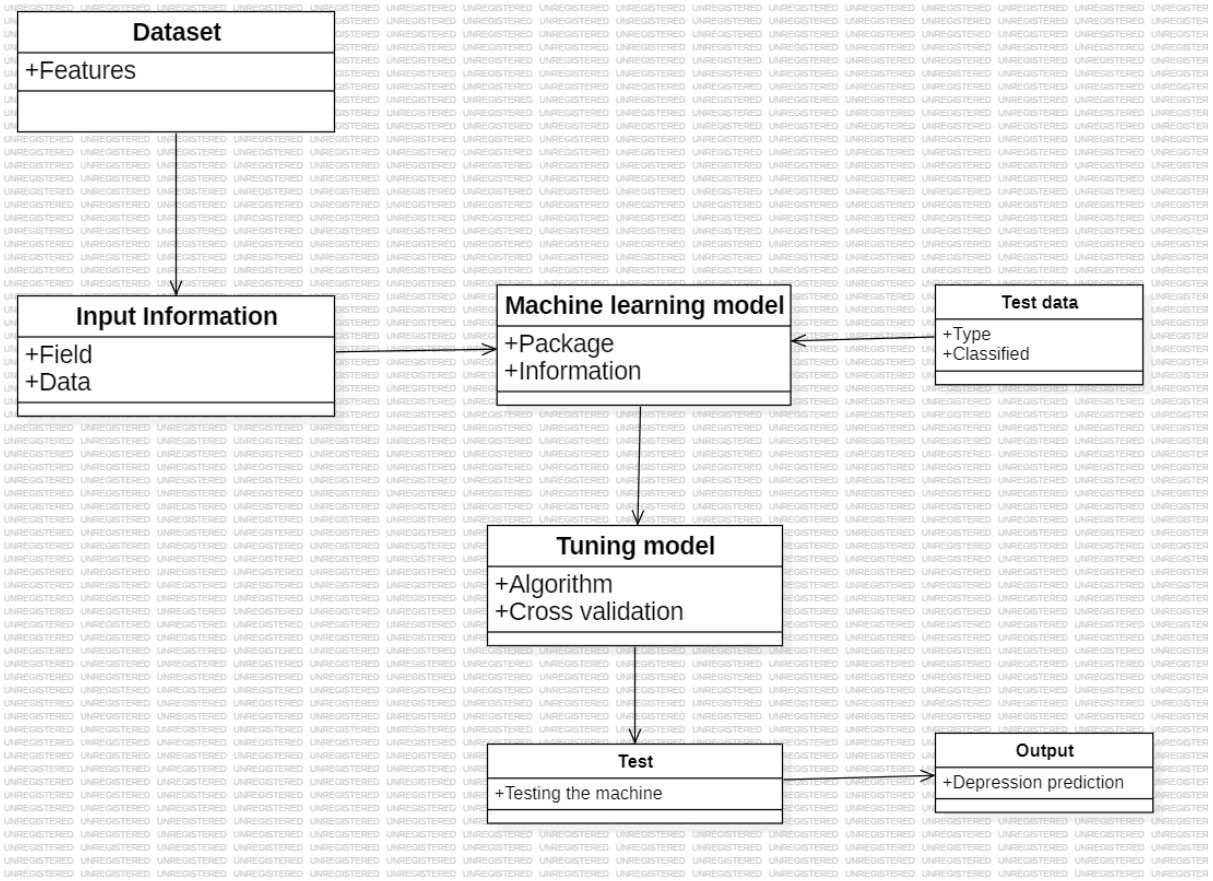


FIG 5.3: CLASS DIAGRAM

5.4 Activity diagram:

This schematic is a more intricate rendition of a sequential diagram illustrating the transfer of data between different tasks. It outlines the synchronization of tasks to provide a service across various levels of abstraction.

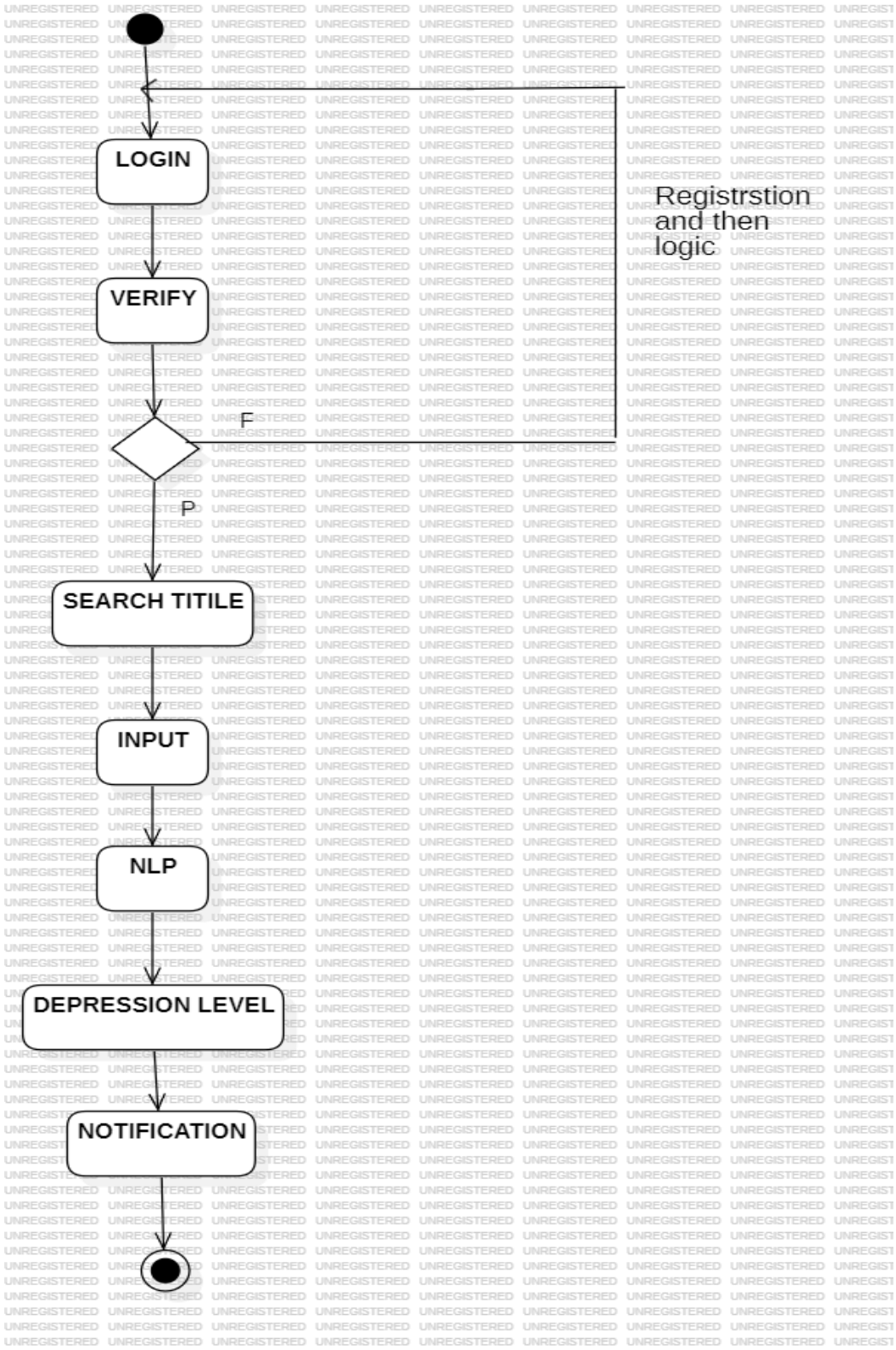


FIG 5.4: ACTIVITY DIAGRAM

5.5 Sequence diagram:

The illustration in this diagram represents the procedures and elements involved in addition to the order of messages exchanged to execute the required functionality. Sequence diagrams are commonly linked with implementing use cases within the 4+1 architectural view model of the evolving system.

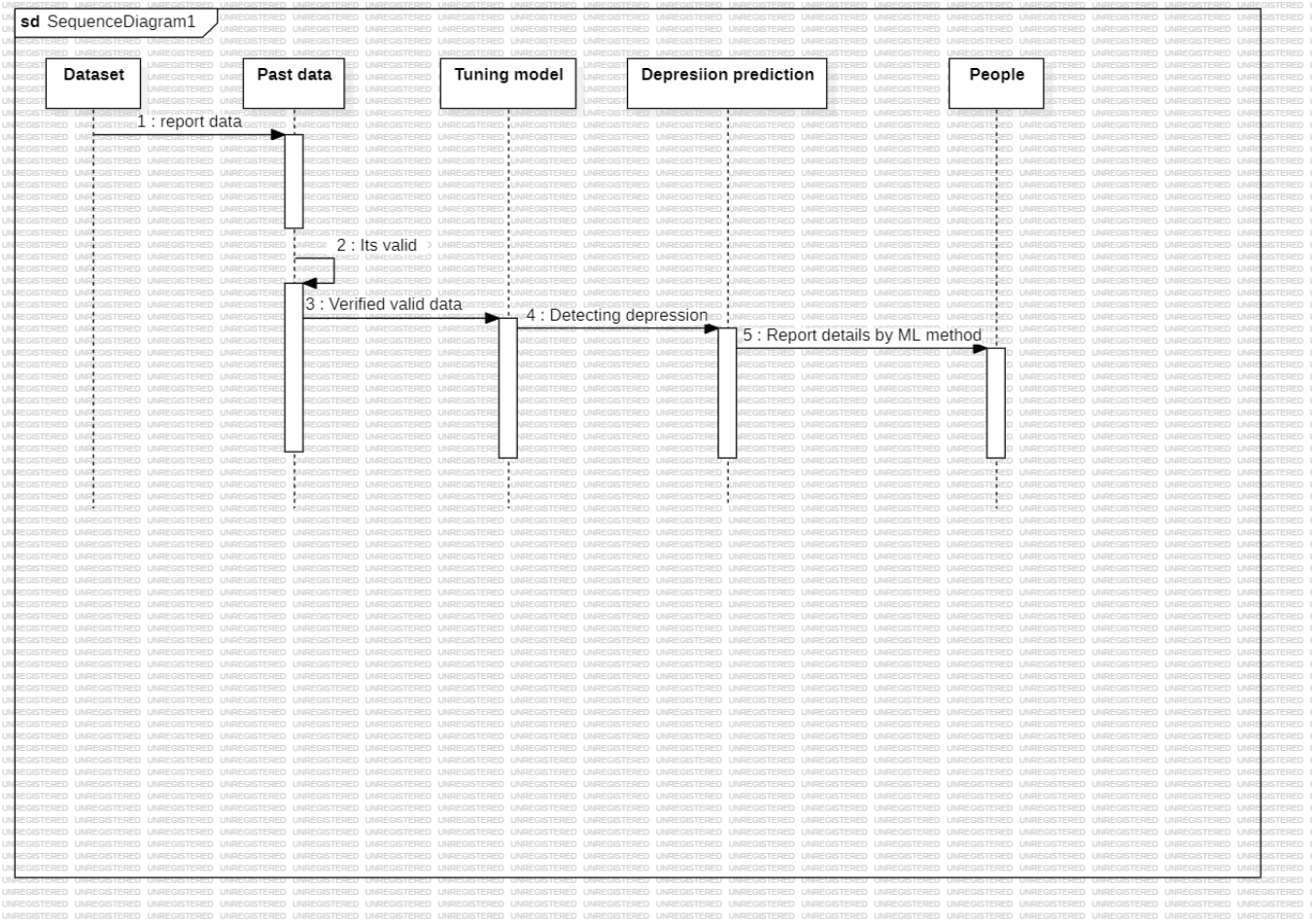


FIG 5.5: SEQUENCE DIAGRAM

5.6 Architecture

The structure of Natural Language Processing (NLP) comprises various essential elements: text preparation (such as tokenization, stemming, and removing stop words), feature extraction (including Bag of Words, TF-IDF, and word embeddings), and constructing models through machine learning techniques (like Naive Bayes, SVM, and neural networks). Analyzing text involves methods like Named Entity Recognition, sentiment assessment, and part-of-speech

identification. When evaluating models, metrics like accuracy and F1-score are employed, followed by deployment. It includes API integration and continuous monitoring. Applications range from text classification and summarization to machine translation and chatbots.

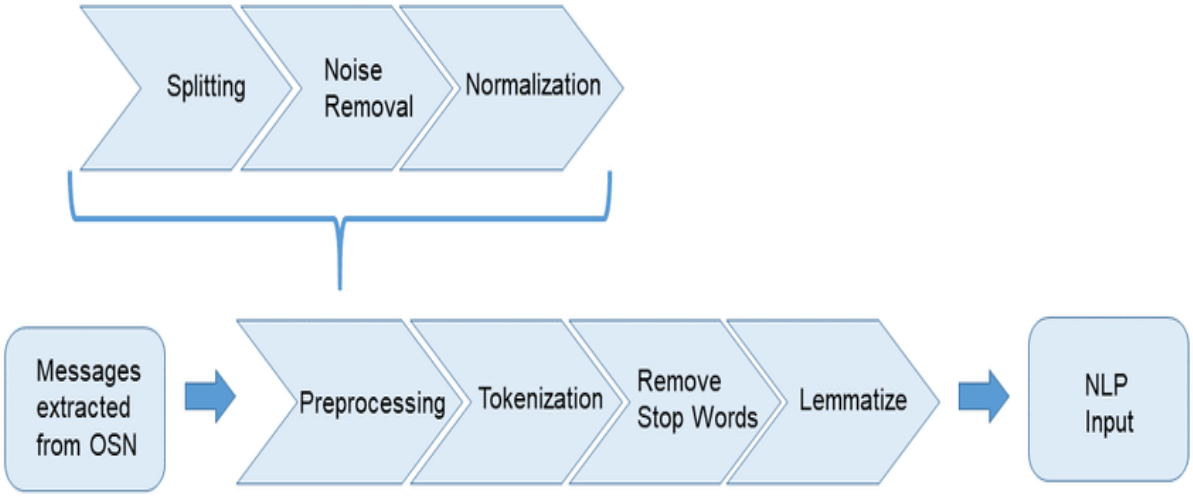
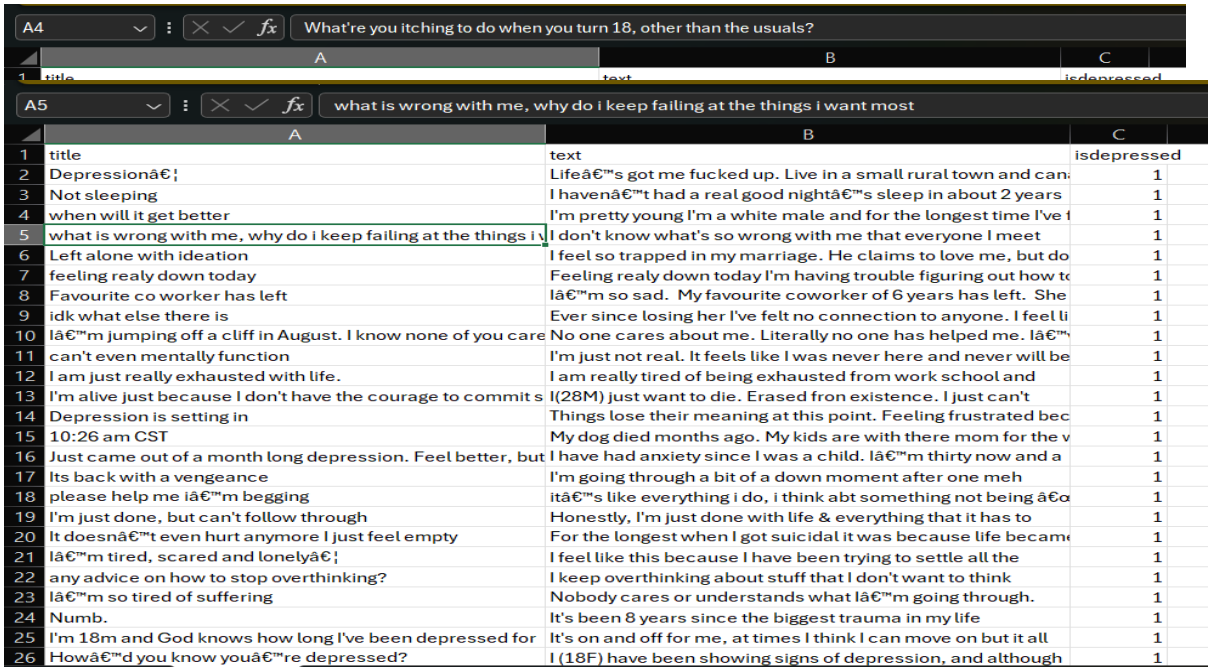


FIG 5.6: NLP ARCHITECTURE

6. IMPLEMENTATION

6.1 Sample Data for Problem Statement

Texts from various social media platforms and user interests are included in the file. These texts consist of both positive tone and negative tone of sentences. They are classified according to their tone into two groups of data as casual and depressed conversation. A few sample texts from the sets are shown below.



A4 What're you itching to do when you turn 18, other than the usuals?

A5 what is wrong with me, why do i keep failing at the things i want most

	A	B	C
1	title	text	isdepressed
2	Depressionâ€¦	Lifeâ€™s got me fucked up. Live in a small rural town and can	1
3	Not sleeping	I havenâ€™t had a real good nightâ€™s sleep in about 2 years	1
4	when will it get better	I'm pretty young I'm a white male and for the longest time I've	1
5	what is wrong with me, why do i keep failing at the things i v	I don't know what's so wrong with me that everyone I meet	1
6	Left alone with ideation	I feel so trapped in my marriage. He claims to love me, but do	1
7	feeling realy down today	Feeling realy down today I'm having trouble figuring out how t	1
8	Favourite co worker has left	Iâ€™m so sad. My favourite coworker of 6 years has left. She	1
9	idk what else there is	Ever since losing her I've felt no connection to anyone. I feel li	1
10	Iâ€™m jumping off a cliff in August. I know none of you care	No one cares about me. Literally no one has helped me. Iâ€™	1
11	can't even mentally function	I'm just not real. It feels like I was never here and never will be	1
12	I am just really exhausted with life.	I am really tired of being exhausted from work school and	1
13	I'm alive just because I don't have the courage to commit s	I(28M) just want to die. Erased fron existence. I just can't	1
14	Depression is setting in	Things lose their meaning at this point. Feeling frustrated bec	1
15	10:26 am CST	My dog died months ago. My kids are with there mom for the v	1
16	Just came out of a month long depression. Feel better, but	I have had anxiety since I was a child. Iâ€™m thirty now and a	1
17	Its back with a vengeance	I'm going through a bit of a down moment after one meh	1
18	please help me iâ€™m begging	itâ€™s like everything i do, i think abt something not being â€¢	1
19	I'm just done, but can't follow through	Honestly, I'm just done with life & everything that it has to	1
20	It doesnâ€™t even hurt anymore I just feel empty	For the longest when I got suicidal it was because life becam	1
21	Iâ€™m tired, scared and lonelyâ€¦	I feel like this because I have been trying to settle all the	1
22	any advice on how to stop overthinking?	I keep overthinking about stuff that I don't want to think	1
23	Iâ€™m so tired of suffering	Nobody cares or understands what Iâ€™m going through.	1
24	Numb.	It's been 8 years since the biggest trauma in my life	1
25	I'm 18m and God knows how long I've been depressed for	It's on and off for me, at times I think I can move on but it all	1
26	Howâ€™d you know youâ€™re depressed?	I (18F) have been showing signs of depression, and although	1

FIG 6.1: SAMPLE DATA FROM THE DATASET

6.2 MODEL CONSTRUCTION

6.2.1 Importing Libraries and Creating Data frames

```
[1]  import pandas as pd
     import numpy as np
```

```
     depresseddf = pd.read_csv('/content/depressed.csv')
     newdepresseddf = pd.DataFrame()
     newdepresseddf["text"] = depresseddf["title"] + " " + depresseddf["text"]
     newdepresseddf['isdepressed'] = depresseddf["isdepressed"]
     newdepresseddf.head()
```

```
[ ]  casualconvdf = pd.read_csv("/content/casualconversation.csv")
     newcasualconvdf = pd.DataFrame()
     newcasualconvdf["text"] = casualconvdf["title"] + " " + casualconvdf["text"]
     newcasualconvdf['isdepressed'] = casualconvdf["isdepressed"]
     newcasualconvdf.head()
```

By importing these libraries, the code is setting up the environment to use the functionalities provided by Pandas and NumPy for data manipulation, analysis, and numerical computations. These libraries are commonly used together in data science and machine learning projects.

The Data Frame will contain all the data from the CSV file, organized into rows and columns.

6.2.2 Merging the Data

```
     # Merging Depressed and Casual Conversation Data
     mixeddf = pd.concat([newdepresseddf,newcasualconvdf])

     # Also Merging both title and text of posts as both contain important text data

     mixeddf.head()
```

6.2.3 Checking data counts

```
[ ]  # Checking depressed(1) and not derpressed(2) counts
     mixeddf['isdepressed'].value_counts()
```

```
isdepressed
1    1000
0    1000
Name: count, dtype: int64
```

6.2.4 Setting up an Environment for text preprocessing

```
import re
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
from bs4 import BeautifulSoup
import unicodedata

nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
```

6.2.5 Text preprocessing

```
def expand_contractions(text):
    # Define your contraction mapping dictionary
    contractions_dict = {"can't": "cannot", "won't": "will not", "i'm": "i am", "isn't": "is not"}
    contractions_re = re.compile('(%s)' % '|'.join(contractions_dict.keys()))

    def replace(match):
        return contractions_dict[match.group(0)]

    return contractions_re.sub(replace, text)

def remove_emails(text):
    return re.sub(r'\S+@\S+', '', text)

def remove_urls(text):
    return re.sub(r'http\S+|www\S+|https\S+', '', text)

def remove_html_tags(text):
    return BeautifulSoup(text, "html.parser").get_text()

def remove_rt(text):
    return re.sub(r'\brt\b', '', text).strip()

def remove_accented_chars(text):
    return unicodedata.normalize('NFKD', text).encode('ascii', 'ignore').decode('utf-8', 'ignore')

def remove_special_chars(text):
    return re.sub(r'[^a-zA-Z0-9\s]', '', text)

def remove_repeated_chars(text):
    return re.sub(r'(.)\1{2,}', r'\1', text)
```

```
[ ] def get_clean(x):
        x = str(x).lower().replace('\\', '').replace('_', ' ')
        x = expand_contractions(x)
        x = remove_emails(x)
        x = remove_urls(x)
        x = remove_html_tags(x)
        x = remove_rt(x)
        x = remove_accented_chars(x)
        x = remove_special_chars(x)
        x = remove_repeated_chars(x)
        return x
```

```
[ ] mixeddf['text'] = mixeddf['text'].apply(lambda x: get_clean(x))
    newcasualconvdf['text'] = newcasualconvdf['text'].apply(lambda x: get_clean(x))
    newdepresseddf['text'] = newdepresseddf['text'].apply(lambda x: get_clean(x))

    mixeddf.head()
```

```
##remove freq word function
def freqremovel(train1, freq):
    train1['text'] = train1['text'].apply(lambda x: " ".join(x for x in x.split() if x in freq))
    return train1
depressedfreq = pd.Series(' '.join(newdepresseddf['text']).split()).value_counts()[220:-600]
newdepresseddf = freqremovel(newdepresseddf,depressedfreq)

casualconvfreq = pd.Series(' '.join(newcasualconvdf['text']).split()).value_counts()[150:-200]
newcasualconvdf = freqremovel(newcasualconvdf,casualconvfreq)
```

6.2.6 Visual Representation of Word Frequency

```
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt


def showwordcloud(df):
    comment_words = ''
    stopwords = set(STOPWORDS)

    for val in df.text:
        val = str(val)
        tokens = val.split()
        for i in range(len(tokens)):
            tokens[i] = tokens[i].lower()

        comment_words += " ".join(tokens)+" "

    wordcloud = WordCloud(width = 800, height = 800,
                    background_color ='black',
                    stopwords = stopwords,
                    min_font_size = 10).generate(comment_words)


    plt.figure(figsize = (8, 8), facecolor = None)
    plt.imshow(wordcloud)
    plt.axis("off")
    plt.tight_layout(pad = 0)

    plt.show()
```

```
print("Word Cloud for Depression Text")
showwordcloud(newdepresseddf)
```

```
print("Word Cloud for Casual Conversation")
showwordcloud(newcasualconvdf)
```

6.3 MODEL TRAINING

```
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.model_selection import train_test_split, cross_validate
from sklearn.svm import LinearSVC
from sklearn.metrics import classification_report

from sklearn.svm import SVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
```

6.3.1 TF-IDF (Term Frequency * Inverse Document Frequency)

```
# Creating Vectorizer With max first 20,000 words from dictionary
# ngram will be uni, bi & tri
# Tokenization will be char by char and not by word (better results)
tfidf1 = TfidfVectorizer(max_features = 20000, ngram_range=(1,3), analyzer='char')

# Tokenization will be Word by Word
tfidf2 = TfidfVectorizer(max_features = 20000, ngram_range=(1,3), analyzer='word')
```

```
[ ] X1 = tfidf1.fit_transform(mixeddf['text'])
    y1 = mixeddf['isdepressed']
    X2 = tfidf2.fit_transform(mixeddf['text'])
    y2 = mixeddf['isdepressed']
```

```
[ ] # Number Of Unique Character features in Total Data
    print("Feature Extraction using Characters")
    print("Total Unique Characters Extracted" ,len(tfidf1.vocabulary_))
    vocabdict = tfidf1.vocabulary_
    {k: vocabdict[k] for k in list(vocabdict)[:50]}
```

```
[ ] # Number Of Unique Word features in Total Data
    print("Feature Extraction using Words")
    print("Total Unique Words Extracted",len(tfidf2.vocabulary_))

    vocabdict = tfidf2.vocabulary_
    {k: vocabdict[k] for k in list(vocabdict)[:50]}
```

```
[ ] print("Extraction by Chars:(Rows,Features)",X1.shape)
    print("Extraction by Words:(Rows,Features)",X2.shape)
```

6.3.2 Split Train and Test Data (80/20)

```
[ ] # Split Train and Test data 80/20
    X1_train, X1_test, y1_train, y1_test = train_test_split(X1,y1,test_size=0.2,random_state=0) # Char Tokenization
    X2_train, X2_test, y2_train, y2_test = train_test_split(X2,y2,test_size=0.2,random_state=0) # Word Tokenization
    print("Number of Depressed(1) and Non-Depressed(0) Test Cases")
    print(y1_test.value_counts())
    print(y2_test.value_counts())
```

6.3.3 Classifiers Used

```
[ ] # List of all the classifiers to be used
    classifiers = [ LinearSVC(),
                    SVC(kernel='rbf',C=10),
                    SVC(kernel='linear',C=10),
                    MultinomialNB(),
                    LogisticRegression(),
                    RandomForestClassifier(),
                    DecisionTreeClassifier()
                  ]
```

6.3.4 Model Score Comparison With TF-IDF and Character Tokenization

Models Used: Linear SVC, SVC, Multinomial Naive Bayes, Logistic Regression, Random Forest Classifiers, Decision Tree Classifiers

```
[ ] for clf in classifiers:
        print(clf,"Average Score ",end="")
        scores=cross_validate(clf,X1, y1,cv=10)
        print('{:.2%}'.format(np.mean(scores["test_score"])))
```

```
LinearSVC() Average Score 89.15%
SVC(C=10) Average Score 88.90%
SVC(C=10, kernel='linear') Average Score 88.80%
MultinomialNB() Average Score 79.85%
LogisticRegression() Average Score 84.90%
RandomForestClassifier() Average Score 86.50%
DecisionTreeClassifier() Average Score 72.40%
```

6.3.5 Model Score Comparison With TF-IDF and Word Tokenization

Models Used: Linear SVC, SVC, Multinomial Naive Bayes, Logistic Regression, Random Forest Classifiers, Decision Tree Classifiers

```
[ ]  for clf in classifiers:
         print(clf,"Average Score ",end="")
         scores=cross_validate(clf,X2, y2,cv=10)
         print('{:.2%}'.format(np.mean(scores["test_score"])))
```

```
LinearSVC() Average Score 89.70%
SVC(C=10) Average Score 89.45%
SVC(C=10, kernel='linear') Average Score 89.65%
MultinomialNB() Average Score 87.60%
LogisticRegression() Average Score 87.90%
RandomForestClassifier() Average Score 86.75%
DecisionTreeClassifier() Average Score 75.85%
```

We got Maximum Accuracy of 90.00% with SVC and TD-IDF Word Tokenization

6.3.6 Classification report for SVC and TF-IDF Word Tokenization

```
[ ]      print("Classification Report For SVC and TD-IDF Word Tokenization")
         model = SVC(C=10,kernel='linear')
         model.fit(X2_train,y2_train)
         y_pred = model.predict(X2_test)
         print(classification_report(y2_test,y_pred))
```

```
Classification Report For SVC and TD-IDF Word Tokenization
              precision    recall  f1-score   support

           0       0.88      0.90      0.89       200
           1       0.89      0.88      0.88       200

    accuracy                           0.89       400
   macro avg       0.89      0.89      0.88       400
weighted avg       0.89      0.89      0.88       400
```

6.4 Let's Test the Model with some Texts

```
def predict_list(predtext):
    for x in predtext:
        x = get_clean(x)
        vec = tfidf2.transform([x])
        print("The Person Is Depressed") if(model.predict(vec)) else print("The Person Is not Depressed")
```

```
[ ] text1 = ["im feeling hopeless, have no energy, cant work. it sucks",
             "its very good to have everything working, i bought a car, got good marks",
             "I Saved My Family's Cat Today",
             "I have a cute dog, started a buisness, went to vacation last week",
             "I am working with facebook, i have good salary, have two chilren, everythings alright",
             "I have too much debt, my wife left, everything is falling apart, cant handle"
             ]

    predict_list(text1)
```

7. SOFTWARE TESTING

Software testing refers to the evaluation conducted before the actual execution of. The primary goal software testing is to ensure the expected output meets the requirements without any errors or defects.

7.1 Unit Testing:

Unit testing in NLP ensures the reliability and correctness of individual components in NLP pipelines. It includes tests on writing preprocessing functions, feature extraction, model training, and predictive functions.Text preprocessingTagging, stopword removal, lead generation and text cleaning functions should be tested. For example, get_clean can be tested to ensure that it correctly expands contractions, removes emails, URLs, HTML tags, special characters, and handles accents and repeating characters.

Feature extraction:

Feature extraction methods such as TfIdf Vectorizer and Count Vectorizer should be tested to verify the dimensions of the results and the expected features. For example, testing the TfidfVectorizer involves checking that the transformed output has the correct shape.

Model Training and Prediction:

Tests should ensure that the model training process works as expected and the predictions are accurate. For example, after running a logistic regression model, unit tests can be used to ensure that the accuracy of the model exceeds a certain threshold.

7.2 Integration Testing:

Following unit testing, integration testing is conducted. This phase involves using the results of unit testing as the input for integrated testing, with a focus on the functional requirements. During this process, individual code units within a module are assembled for testing.

Integration testing for ship detection of remote sensing images involves scrutinizing and combining various parts/modules of the system. It encompasses validating input/output mechanisms, assessing integration performance, and ensuring seamless integration of preprocessing and feature extraction components. Furthermore, it entails verifying the accurate integration of machine learning models or rule-based algorithms with other system elements. The primary goal is to ensure the synergy of components, accurate data transfer among modules, and the achievement of desired outcomes.

Through integration testing, potential issues or inconsistencies in the integration process can be identified and rectified, leading to the development of a dependable and efficient ship detection system.

7.3 Acceptance Testing:

Acceptance trying out is achieved with the consequences of system trying out as a starting point. this is completed to check that the anticipated and assumed necessities suit.

Examining the ship detection system to make sure it satisfies the intended requirements and acceptance criteria is the main goal of acceptance testing for ship detection utilising remote sensing pictures. During this testing step, the system's overall performance, including its correctness, dependability, and robustness, are evaluated. It might entail putting the system to the test using a variety of real-world remote sensing photos shot under various circumstances and settings. The objective is to confirm that the system can accurately and efficiently detect ships, reduce false positives, manage a variety of ship sizes and orientations, and work consistently in a variety of geographical locations and weather circumstances. Acceptance testing assists in ensuring that the ship detection system satisfies end-user requirements and is prepared for implementation in practical applications.

7.4 Testing on our System:

After integrated testing, machine checking out is finished. As a end result, each purposeful and nonfunctional trying out are included in the process. The incorporated checking out output is used as the input for system checking out. This checking out is accomplished on the gadget's design or behavior.

8.RESULTS

```
[ ]  def predict_list(predtext):
         for x in predtext:
             x = get_clean(x)
             vec = tfidf2.transform([x])
             print("The Person Is Depressed") if(model.predict(vec)) else print("The Person Is not Depressed")
```

```
[ ]  text1 = ["im feeling hopeless, have no energy, cant work. it sucks",
              "its very good to have everything working, i bought a car, got good marks",
              "I Saved My Family's Cat Today",
              "I have a cute dog, started a buisness, went to vacation last week",
              "I am working with facebook, i have good salary, have two chilren, everythings alright",
              "I have too much debt, my wife left, everything is falling apart, cant handle"
              ]

     predict_list(text1)
```

```
The Person Is Depressed
The Person Is not Depressed
The Person Is not Depressed
The Person Is not Depressed
The Person Is not Depressed
The Person Is Depressed
```

FIG 8.1: OUTPUT OF FINAL MODEL

Model: SVC

Classifiers used:

SVC

Multinomial NB

Logistic Regression

Random Forest Classifier

Decision Tree Classifier

Accuracy: 90%

9.CONCLUSION

9.1Conclusion

The depression prediction project utilizing NLP and machine learning has demonstrated the feasibility of detecting depressive symptoms from social media text with high accuracy. By leveraging techniques like TF-IDF, CountVectorizer, and multiple classifiers, the system effectively identifies depressive posts, aiding in early intervention efforts.

9.2Future Enhancements

Future work could improve accuracy and generalizability by incorporating advanced deep learning models like BERT or GPT. Additionally, expanding the dataset to include diverse linguistic and cultural contexts can enhance the system's robustness. Real-time monitoring and integration with mental health support platforms could provide timely assistance to individuals in need.

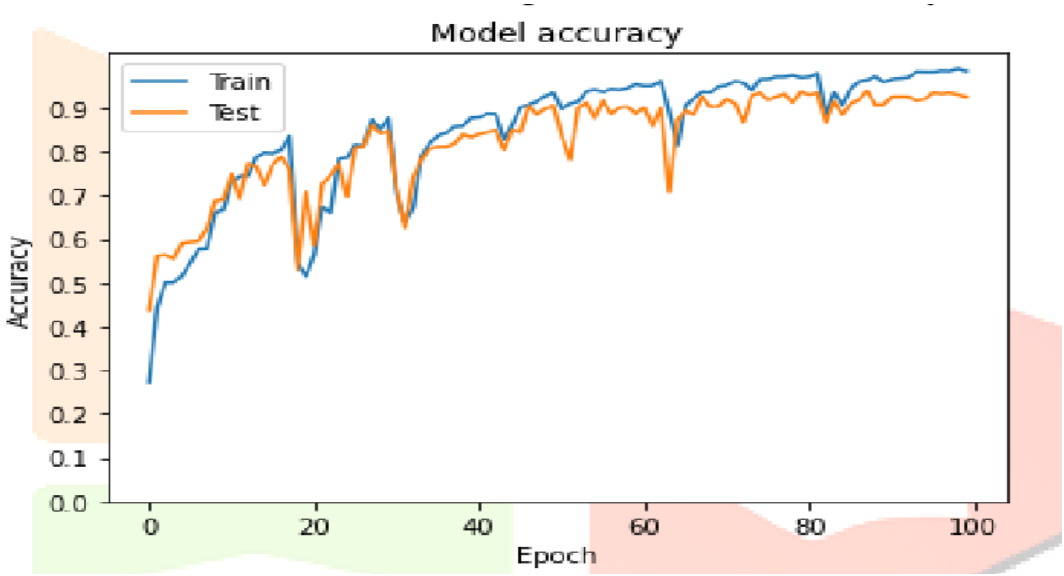


FIG 9: ACCURACY OF THE MODEL

10.BIBLOGRAPHY

[1] Depression detection from Twitter posts using NLP and Machine learning techniques
Shreyas S Korti; Suvarna G Kanakaraddi

[2] Depression Detection from social media Textual Data Using Natural Language Processing and Machine Learning techniques
Farhan Kabir; Md. Ali Hossain; A. F. M. Minhazur Rahman; Sadia Zaman Mishu

[3] Recognizing Suicidal Intent in Depressed Population using NLP: A Pilot Study
Samiha Binte Hassan, Sumaiya Binte Hassan

[4] Depression detection using emotion artificial intelligence through NLP
Mandar Deshpande; Vignesh Rao

[5] Depression and Suicide Analysis Using Machine Learning and NLP
Pratyaksh Jain, Karthik Ram Srinivas

[6] https://scikit-learn.org

[7] https://www.researchgate.net/publications/374148247

[8] https://pypi.org

[9] https://star-uml.io

[10] Natural language processing applied to mental illness detection: a narrative review
Tianlin Zhang, Annika M.Schoene

[11] Assessing ML Classification Algorithms and NLP Techniques for Depression Detection
Giuliano Lorenzoni, Cristina Tavares, Nathalia Nascimento