SENTIMENT ANALYSIS FOR DEPRESSION DETECTION FROM SOCIAL MEDIA TEXTS

A Project Report submitted in partial fulfilment of the requirement forthe Award of the Degree of

BACHELOR OF ENGINEERING

In

ARTIFICIAL INTELLIGENCE & DATA SCIENCE

By

M. VINAY (160720747030)

SYED IBRAHIM SALEEM (160720747001)

J. AJAY (160720747026)

Under the Guidance of

Mr. V. VENKATRAM,

Assistant Professor, Dept. of CSE

Department of Artificial Intelligence & Data Science

Methodist College of Engineering and Technology, King Koti, Abids, Hyderabad-500001.

2023-2024

SENTIMENT ANALYSIS FOR DEPRESSION DETECTION FROM SOCIAL MEDIA TEXTS

A project report submitted in partial fulfillment of the requirement for the Award of the Degree of

BACHELOR OF ENGINEERING

In

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

By

M. VINAY (160720747030)

SYED IBRAHIM SALEEEM (160720747001)

J. AJAY (160720747026)

Under the Guidance of
Mr. V. VENKATRAM,
Assistant Professor, Dept. of CSE

Department of Artificial Intelligence & Data Science

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001.

2023-2024

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001,

Department of Artificial Intelligence & Data Science

DECLARATION BY THE CANDIDATES

We, **M. VINAY (160720747030), SYED IBRAHIM SALEEM (160720747001)** and **J. AJAY (160720747026)** students of Methodist College of Engineering and Technology, pursuing Bachelor's degree in Artificial Intelligence and Data Science, hereby declare that this Project report entitled **"SENTIMENT ANALYSIS FOR DEPRESSION DETECTION OF SOCIAL MEDIAL TEXTS",** carried out under the guidance of **Mr. V. VENKATRAM** submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Artificial Intelligence and Data Science. This is a record work carried out by us and the results embodied in this report have not been reproduced/copied from any source.

M. VINAY (160720747030)

SYED IBRAHIM SALEEM (160720747001)

J. AJAY (160720747026)

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001.

Department of Artificial Intelligence & Data Science

CERTIFICATE BY THE SUPERVISOR

This is to certify that this Dissertation on project work entitled "**SENTIMENT ANALYSIS FOR DEPRESSION DETECTION FROM SOCIAL MEDIA TEXTS**" *by* **M. VINAY (160720747030), SYED IBRAHIM SALEEM (160720747001) AND J. AJAY (160720747001)** submitted in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Artificial Intelligence and Data Science, during the academic year 2023-2024, is a bonafide record ofwork carried out by them.

Date:

Mr. V. VENKATRAM,
Assistant Professor,
Dept. of CSE.

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001.

Department of Artificial Intelligence & Data Science

CERTIFICATE BY HEAD OF THE DEPARTMENT

This is to certify that this Dissertation on project work entitled "**SENTIMENT ANALYSIS FOR DEPRESSION DETECTION FROM SOCIAL MEDIA TEXTS**" *by* **M. VINAY (160720747030), SYED IBRAHIM SALEEM (160720747001) AND J. AJAY (160720747026)** submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Artificial Intelligence and Data Science of the Osmania University, Hyderabad, during the academic year 2023-2024, is a bonafide record of work carried out by them.

Dr. P. LAVANYA,
Professor &
Head of the Department

Date:

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001.

Department of Artificial Intelligence & Data Science

PROJECT APPROVAL CERTIFICATE

This is to certify that this Dissertation on project work entitled "**SENTIMENT ANALYSIS FOR DEPRESSION DETECTION OF SOCIAL MEDIA TEXTS**" *by* **M. VINAY (160720747030), SYED IBRAHIM SALEEM (160720747001) AND J. AJAY (160720747026),** submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Artificial Intelligence and Data Science of the Osmania University, Hyderabad, during the academic year 2023-2024, is a bonafide record of work carried out by them.

INTERNAL	**EXTERNAL**	**HOD**

ACKNOWLEDGEMENTS

We would like to express our sincere gratitude to my Project guide **Mr. V. Venkatram, Assistant Professor, CSE,** for giving us the opportunity to work on this topic. It would never be possible for us to take this project to this level without his innovative ideas and his relentless support and encouragement.

We would like to thank our project coordinator **Mr. V. Venkatram, Assistant Professor, CSE,** who helped us by being an example of high vision and pushing towards greater limits of achievement.

Our sincere thanks to **Dr. P. Lavanya, Professor** and **Head of the Department of Computer Science and Engineering,** for her valuable guidance and encouragement which has played a major role in the completion of the project and for helping us by being an example of high vision and pushing towards greater limits of achievement.

We would like to express a deep sense of gratitude towards the **Dr. Prabhu G. Benakop, Principal, Methodist College of Engineering and Technology,** for always being an inspiration and for always encouraging us in every possible way.

We would like to express a deep sense of gratitude towards the **Dr. Lakshmipathi Rao, Director, Methodist College of Engineering and Technology,** for always being an inspiration and for always encouraging us in every possible way.

We are indebted to the Department of Computer Science & Engineering and Methodist College of Engineering and Technology for providing us with all the required facility to carry our work in a congenial environment. We extend our gratitude to the CSE Department staff forproviding us to the needful time to time whenever requested.

We would like to thank our parents for allowing us to realize our potential, all the support they have provided us over the years was the greatest gift anyone has ever given us and also for teaching us the value of hard work and education. Our parents have offered us with tremendous support and encouragement, thanks to our parents for all the moral support and theamazing opportunities they have given us over the years.

Department of Computer Science & Engineering
Vision & Mission

VISION

To become a leader in providing Computer Science & Engineering education with emphasis on knowledge and innovation.

MISSION

M1: To offer flexible programs of study with collaborations to suit industry needs
M2: To provide quality education and training through novel pedagogical practices
M3: **To** Expedite high performance of excellence in teaching, research and innovations.
M4: To impart moral, ethical valued education with social responsibility.

Program Educational Objectives

Graduates of Compute Science and Engineering at Methodist College of Engineering and Technology will be able to:

PEO1: Apply technical concepts, Analyze, synthesize data to Design and create novel products and solutions for the real-life problems.

PEO2: Apply the knowledge of Computer Science Engineering to pursue higher education with due consideration to environment and society.

PEO3: Promote collaborative learning and spirit of team work through multidisciplinary projects

PEO4: Engage in life-long learning and develop entrepreneurial skills.

Program Specific Outcomes

At the end of 4 years, Compute Science and Engineering graduates at MCET will be able to:

PSO1: Apply the knowledge of Computer Science and Engineering in various domains like networking and data mining to manage projects in multidisciplinary environments.

PSO2: Develop software applications with open-ended programming environments**.**

PSO3: Design and develop solutions by following standard software engineering principles and implement by using suitable programming languages and platforms

PROGRAM OUTCOMES

PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

ABSTRACT

With the increase in mental health awareness, various social media platforms have become potential sources for text-based information gathering on people's emotional well-being. However, it may be hard to obtain accurate results due to the informal nature of such texts which are often characterized by slang and abbreviations among others hence causing the provided data to be biased. The primary objective of this study is to identify signs of depression in social media posts by building an LSTM model that can analyze individuals' messages. Several stages are involved in this process such as preprocessing the dataset, extracting features from it, and training the model among others. Also, a labeled set of social media texts will be used to train the model while its performance will be assessed using different measures like accuracy, precision, recall etcetera. The project findings could help professionals in the mental health care sector recognize potential risks among users based on their online activities while at the same time providing insights into various patterns linked with self-destructive behaviors on such platforms to come up with suitable intervention strategies.

TABLE OF CONTENTS

1. INTRODUCTION 1

1.1. SENTIMENT ANALYSIS 1

1.2 IMPORTANCE 1

1.3 TYPES 1

1.4. WORKING 5

1.5. APPROACHES 6

1.6 DEPRESSION 7

1.7. SYMPTOMS OF DEPRESSION 8

1.8. CAUSES OF DEPRESSION 9

1.9. TYPES OF DEPRESSION 10

1.10 TREATMENT 10

1.11 SOCIAL MEDIA’S CONTRIBUTION 11

1.12 USE OF SENTIMENT ANALYSIS TO DETECT DEPRESSION 13

2. LITERATURE REVIEW 15

2.1 LITERATURE SURVEY 15

2.2 GAP ANALYSIS 28

2.3 OBJECTIVES 28

2.4 PROBLEM STATEMENT 29

3. DESGIN ANALYSIS 30

3.1 MODEL ARCHITECTURE 30

3.1.1 DATA COLLECTION 31

3.1.2 DATA PRE-PROCESSING 31

1.9.3 FEATURE EXTRACTION 33

1.9.4 TRAINING 34

1.9.5 EVALUATION 35

4. MODULES AND IMPLEMENTATION 37

4.1 DATA COLLECTION 37

4.2 PRE-PROCESSING 38

4.3 DATA CONVERSION 39

4.4 WORD TOKENIZATION 41

4.5 DATASET CLEANING 42

4.6 FEATURE EXTRACTION 44
4.7 DATA SPLITTING 44
4.8 BUILDING MODEL 46
4.8.1 EMBEDDING LAYER 47
4.8.2 CONVOLUTIONAL LAYER 47
4.8.3 LSTM LAYER 48
4.9 MODEL COMPILATION 49
4.10 DEEP LEARNING 49
4.10.1 ABOUT DEEP LEARNING 49
4.10.2 HOW DOES DEEP LEARNING WORK 50
4.10.3 NEURAL NETWORKS AND ITS TYPES 52
5. TRAINING THE MODEL 62
5.1. EVALUATION METRICS 63
5.1.1 ACCURACY 63
5.1.2 PRECISION 63
5.1.3 RECALL 63
5.1.4. F1 SCORE 63
6. RESULT AND ANALYSIS 65
6.1 EVALUATION METRICS 65
7. SCREENSHOTS 76
8. RESULTS AND ANALYSIS 66
9. CONCLUSION 68
10. REFERENCES 69
APPENDIX 71
A.SAMPLE CODE 71
B. SOFTWARE AND HARDWARE REQUIREMENTS 78
C.TECHNOLOGY USED 79
RESOURCES 80

LIST OF FIGURES

Fig 1.3: Types of Sentiment Analysis 3
Fig 1.8: Signs of Depression 8
Fig 3.1: Model Architecture 27
Fig 4.1.1: depressed Tweets 33
Fig 4.1.2: Random Tweets 34
Fig 4.2.1: Word2vec 34
Fig 4.2.2: Expand Contractions Function 35
Fig 4.3: Cleaning Dataset 35
Fig 4.5: Tokenization 38
Fig 4.6: Embedding Matrix 39
Fig 4.7.1: Splitting the Dataset 39
Fig 4.7.2: Combining the datasets 40
Fig 4.7.3: Shuffling the data 40
Fig 4.8.1: Embedding Layer 41
Fig 4.8.2: Convolutional Layer 41
Fig 4.8.1: LSTM Layer 42
Fig 4.9: Compile the Model 43
Fig 4.10.1: Subsets of Artificial Intelligence 44
Fig 4.10.3: Structure of Neural Networks 45
Fig 4.10.4.1: Structure of Recurrent Neural Networks 46
Fig 4.10.4.2.1: An Example of a Simple Filter Depicting a Curved Line 48
Fig 4.10.4.2.2: Depicts how Convolution Works 49
Fig 4.10.4.2.3: A Padding of 0's around the actual matrix 50
Fig 4.10.4.2.4: This Implementation with stride 2 shows the particular cells that will be Convolved 50
Fig 4.10.4.2.5: Graph of the RELU Function 51
Fig 4.10.4.2.6: An Example of both Max-Pooling and Average-Pooling 51
Fig 4.10.4.2.7: Stacking the concepts under one Roof 52
Fig 4.10.4.2.8: Flow of Data through the Model 53
Fig 4.10.4.2.9: Example of CNN-LSTM Model 54
Fig 5.0.1: Early Stopping 55
Fig 5.0.2: Training the Model 55
Fig 5.1.1: Model Accuracy 56
Fig 6.1: Evaluation Metrics 57
Fig 6.1.1: Training and Validation Accuracy 57
Fig 6.1.2: Training and Testing Loss 57
Fig 7.1: Importing SentimentIntensityAnalyzer 58
Fig 7.2.1: Printing 'Sentiment' 58
Fig 7.2.2: Printing 'Sentiment1' 59
Fig 7.2.3: Printing 'Sentiment2' 60

LIST OF TABLES

Table 2.1: Literature Survey 39

1. INTRODUCTION:

1.1 Sentiment Analysis:

- Sentiment analysis is the extraction of subjective information from text. It is aimed at determining the emotional tone expressed in a passage of writing, whether it is positive, negative, neutral, or even more complex attitudes.
- Today, businesses deal with large volumes of text data such as emails, customer support chat transcripts social media comments, and reviews. Sentiment analysis tools can automatically scan through this text to determine the author's attitude toward a topic.

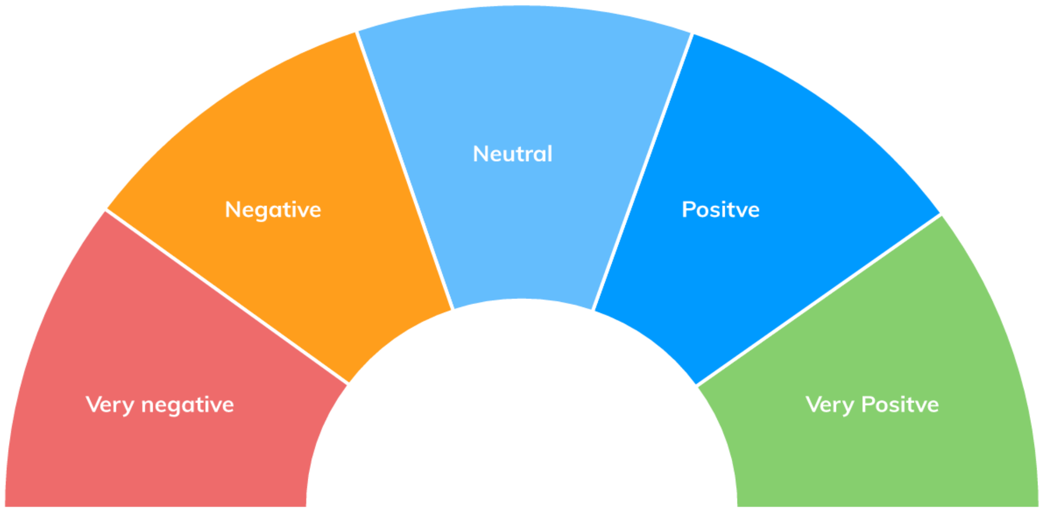


Fig 1.1 Sentiments

1.2 Importance of Sentiment Analysis:

Insights and Satisfaction within Business:

1 **Feedback from Clients:**

Businesses can comprehend customer feedback through reviews, surveys, and social media by utilizing sentiment analysis. Positive and negative feelings are identified for companies to enhance their products, services, and the general experience of the client.

2 **Monitoring Brands**:

Organizations track feelings about their brands to control their reputations and detect potential crises early enough. This enables them to manage public perception proactively as well as maintain the image portrayed by their brands.

3 **Product Development**:

Sentiment analysis insights might assist in product development by indicating the features that customers love and hate thus enabling firms to concentrate on areas that are more important for the users.

4 **Marketing and Market Research:**

Targeted Marketing: Knowing how clients feel helps companies in creating emotionally appealing adverts which can lead to increased sales through higher consumer engagement levels.

5 **Competitive Analysis:**

When conducting competitor analysis, it is important to consider their products' sentiments regarding strengths or weaknesses so as not only positioning yours better but also capitalizing on what they lack.

6 **Healthcare and Mental Health**:

Mental Health Monitoring: To recognize early signs of depression, and anxiety among other mental illnesses social media posts need to be analyzed for any indications showing such states of mind.

7 **Patient Feedback**:

In the academic setting and research field:

- Linguistic research: How emotions are expressed in text has been advanced through sentiment analysis in computational linguistics.
- Behavioral studies: By analyzing sentiment within large datasets analysts can explore human response and behavior in different situations.

1.3 Types of Sentiment Analysis:

Sentiment analysis is concerned with determining the polarity of a given piece of writing – positive, negative, or neutral. However, it also extends beyond this scope to identify specific emotions such as happiness sadness anger etcetera urgency (urgent not urgent) even intentions (interested, not interested).

Depending on how you want to interpret customer feedback and queries, you can define and tailor your categories to meet your sentiment analysis needs.

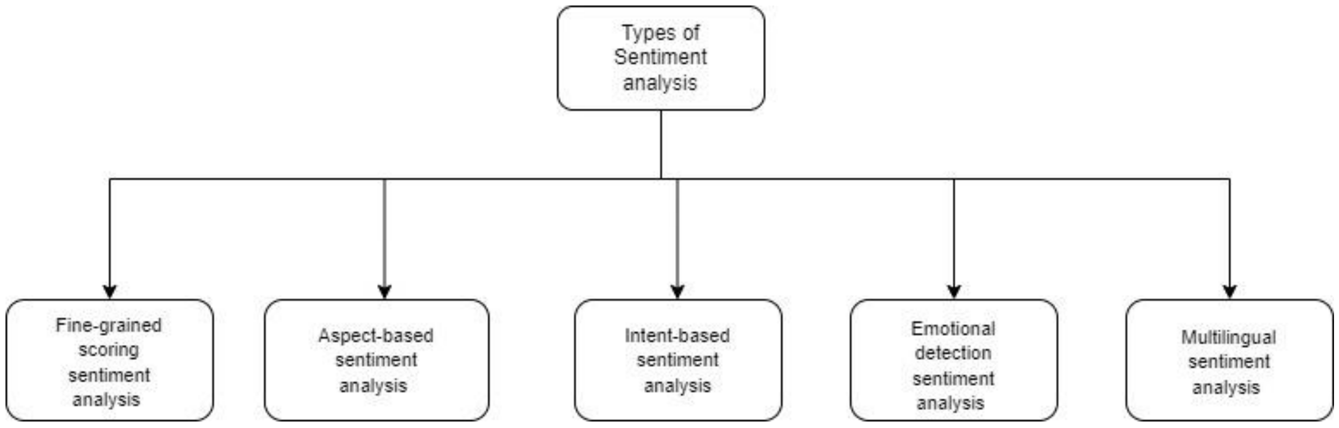

Types of Sentiment analysis
Fine-grained scoring sentiment analysis
Aspect-based sentiment analysis
Intent-based sentiment analysis
Emotional detection sentiment analysis
Multilingual sentiment analysis

Fig 1.3: Types of Sentiment Analysis

Types of Sentiment Analysis:

1. Fine-Grained Scoring Sentiment Analysis
2. Aspect-Based Sentiment Analysis
3. Intent-Based Sentiment Analysis

4. Emotional detection Sentiment Analysis
5. Multilingual Sentiment Analysis

1. Fine-Grained (graded) Sentiment Analysis:

Fine-grained, or graded, sentiment analysis is a type of sentiment analysis that groups text into different emotions and the level of emotion being expressed. The emotion is then graded on a scale of zero to 100, similar to the way consumer websites deploy star-ratings to measure customer satisfaction.

2. Aspect-Based (ABSA) Sentiment Analysis:

Aspect-based sentiment analysis (ABSA) narrows the scope of what's being examined in a body of text to a singular aspect of a product, service or customer experience a business wishes to analyze. For example, a budget travel app might use ABSA to understand how intuitive a new user interface is or to gauge the effectiveness of a customer service Chatbot. ABSA can aid organizations in gauging the relative success or failure of their items according to client satisfaction levels.

3. Intent-Based Sentiment Analysis:

When carrying out market research, companies may use intent-based analysis to determine how clients feel about a product or service. Marketers employ opinion mining to find out where in the buying cycle a particular group of customers stands. They launch specific promotions directed at individuals who have shown interest in making purchases by identifying such phrases as discounts, offers, and reviews during monitored conversations.

4. Emotional Detection Sentiment Analysis:

Emotion detection sentiment analysis aims at understanding the psychological state of the author of a text, as well as the intention, considering the mood at the time of writing. More complicated than both fine-grained and ABSA, it is mostly used to interpret why a person feels something. Instead of using positive, negative, or neutral polarities, this type of sentiment analysis can detect particular feelings like annoyance, apathy, anxiety, and surprise from a text

5. Multilingual Sentiment Analysis:

It can be tough to do multilingual sentiment analysis. A lot of work is required in preprocessing and resources. Most sentiment lexicons are available on the internet for instance while others like translated corpora or noise detection algorithms have to be developed which means you need coding skills if you want to use them.

1.4 How does Sentiment Analysis Work

Sentiment Analysis usually goes through several steps before it can generate the final output.

1 **Preprocessing**:

- Sentiment Analysis recognizes keywords that bring out the main idea in the text during this stage.
- Tokenization involves dividing a sentence into smaller parts known as tokens.
- Lemmatization gets words to their basic form, for example, the basic form of am is changing.
- Stop-word removal sieves off words that do not contribute any meaning to the sentence, for example with, for, at, and of among others.

2 **Keyword Analysis:**

- After the keywords have been identified, NLP technologies can be used to further analyse or rank them according to sentiment. This is done by giving each keyword a score based on how positively or negatively it is likely to be viewed in context, known as the keyword's 'sentiment score'. A sentiment score measures emotions within sentiment analysis systems, providing analysts with an additional perspective when interpreting the text.

1.5 Approaches to Sentiment Analysis:

There are various algorithms to choose from when working on sentiment analysis models, depending on the volume of data you want to analyse and how accurate you want your model to be.

1. **Rule-Based Approach**:

The rule-based method involves identifying, categorizing, and scoring specific keywords using predetermined lexicons. Lexicons are collections of words that signify the author's purpose, emotion, and mood. Marketers attribute sentiment values to positive and negative lexicons to indicate the varying degrees of intensity portrayed by different statements. To predict whether a sentence is positive, negative, or neutral, the program searches for words from the lexicon and totals their corresponding sentiment values. The final score is then compared with the sentiment thresholds to establish the overall emotional orientation.

An Example of Rule-Based Analysis:

In a given system, words such as happy, affordable, and fast are positive lexicon words, while poor, expensive, and difficult are negative ones. Marketers have said that positive word scores range from 5 to 10 and negative word scores range from -1 to -10. There are specific rules designed to detect double negatives (e.g., not bad) as a positive sentiment. It is determined by marketers that an overall sentiment score greater than 3 will be considered positive; between -3 and 3 are mixed feelings.

Advantages and Disadvantages:

The setup of a rule-based sentiment analysis system is simple, but it can be hard to scale. For instance, whenever new keywords that show intention in the input text are found, this will require you to keep adding to your lexicons. This could lead to inaccurate results when dealing with sentences that have been shaped by various cultures.

1. **Machine Learning**:

One way of performing this is by using a process that involves training computer software to identify emotional sentiment from text using machine learning methods coupled with sentiment classification algorithms like neural networks and deep learning.

Training:

During training, data scientists work with large numbers of examples contained in datasets for sentiment analysis. The program uses these as input while teaching itself to conclude. Different word arrangements affect the final sentiment score, so it differentiates them by training on diverse examples in great numbers.

Advantages and Disadvantages:

Machine learning sentiment analysis has been beneficial since it can accurately process a wide range of textual information. The software requires to be trained with enough examples for it to predict the emotional tone of the messages precisely; this is why machines undergoing machine learning are trained. However, a trained model is only specific to a certain business domain. As a result, without retraining, SA software, which has been trained using marketing data, cannot be employed in social media monitoring.

2. **Neural Network**:

Neural networks have gone through exponential growth over the past few years. This technique uses artificial neural networks that classify text as positive, negative, or neutral by working in a similar way to the human brain's structure. Recurrent neural networks (RNNs), such as long short-term memory (LSTM) and gated recurrent units, are some of them that process sequential data.

3. **Hybrid Approach**:

This is a combination of machine learning and rule-based systems that work together to provide the contextual intent of a text. This method uses features from both types of systems so that it can be faster but still accurate; however, merging these two different methods needs time as well as technical effort.

1.6Depression:

Depression, also known as major depressive disorder (MDD), is a common and serious medical illness that negatively affects how you feel, the way you think, and how you act. It causes feelings of sadness and/or a loss of interest in activities once enjoyed. It can lead to a variety of emotional and physical problems and can decrease a person's ability to function at work and at home.

1.7 Symptoms of Depression:

1. **Emotional Symptoms**:

    - Persistently sad, anxious, or "empty" mood.
    - Feelings of hopelessness or pessimism.
    - Feelings of guilt, worthlessness, or helplessness.
    - Loss of interest or pleasure in hobbies and activities.

2. **Physical Symptoms**:

    - Decreased energy or fatigue.
    - Difficulty concentrating, remembering things, or making decisions.
    - Insomnia, early-morning awakening, or oversleeping.
    - Appetite and/or weight changes.
    - Physical aches or pains, headaches, cramps, or digestive problems without any clear cause cannot be explained by a physical illness.

3. **Behavioural Symptoms:**

    - Not spending time with others or being alone all the time.
    - Not doing well at work or school like before.
    - Forgetting to do things or take care of oneself.
    - I was thinking about suicide.

1.8 Causes of Depression:

Depression is not the result of a single cause. Instead, the disorder stems from a combination of factors, such as:

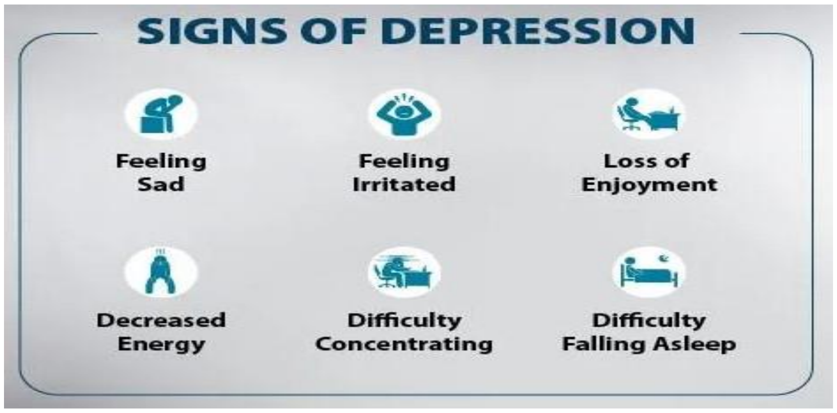


Fig 1.8: Signs of Depression

1. **Biological Causes of Depression:**

   - Genetics: Having a family history of the illness can make it more likely for an individual to develop depression at some point in their life.
   - Brain Chemistry: It has been found that an imbalance in neurotransmitters within the brain can lead to significant depressive states. This is why many antidepressants work by altering the levels of these chemicals.
   - Hormones: Variations or imbalances in hormone levels, particularly those related to such processes as puberty, pregnancy, and menopause, among others, can also contribute to triggering this mental health issue.
   - Medical Conditions: Chronic illnesses, pain, or certain medications can contribute to depression.

2. **Psychological Factors:**

   - Personality Traits: Low self-esteem, dependency, self-criticism, and pessimism are some of the personality traits that may make an individual more susceptible to depression.
   - Trauma and Stress: Dealing with abuse, loss of loved ones, complicated relationships, or financial difficulties could result in someone developing depression.

3. **Environmental Factors:**

- Life Events: Moving, losing one's employment, and splitting up with close people.
- Social Surroundings: The absence of supportive communities, being alone, or living a solitary life.
- Drug and Alcohol Abuse: The signs of depression become more severe when a person is addicted either to alcohol or drugs.

1.9 Types of Depression:

1. **Major Depressive Disorder (MDD):**

   - Severe symptoms that interfere with daily life.

2. **Persistent Depressive Disorder (PDD):**

   - A depressed mood that lasts for at least two years.
   - PDD can have episodes of major depression along with periods of less severe symptoms.

3. **Bipolar Disorder:**

   - Includes mood episodes that range from depressive lows to manic highs.

4. **Postpartum Depression:**

   - Major depression that occurs after childbirth.

5. **Seasonal Affective Disorder (SAD):**

   - Depression typically occurs during the winter months, when there is less natural sunlight.

6. Psychotic Depression:

- Severe depression is accompanied by some form of psychosis, such as delusions or hallucinations.

1.10 Treatments for Depression:

1. **Medication**:

- Antidepressants: monoamine oxidase inhibitors (MAOIs), tricyclic antidepressants, serotonin-norepinephrine reuptake inhibitors (SNRIs), selective serotonin reuptake inhibitors (SSRIs).
- Mood stabilizers: especially for bipolar disorder.
- Antipsychotics: When depressive symptoms are accompanied by delusions or hallucinations.

2. **Psychotherapy**:

- Cognitive Behavioural Therapy (CBT): Focuses on switching pessimistic thoughts.
- Interpersonal therapy (IPT) deals with connections between individuals.
- Psychodynamic therapy delves into unaware emotions and previous events.
- Group therapy provides other people's help who have undergone similar situations.

3. **Lifestyle Changes and Self-Care**:

- Regular Exercise: Physical activity can help improve mood.
- Healthy Diet: Nutritional support for overall well-being.
- Sleep Hygiene: Establishing a regular sleep routine.
- Stress Management: Techniques such as mindfulness, meditation, and relaxation exercises.

4. **Coping and Support**:

- Education: Learning about depression can empower individuals and their loved ones.
- Support groups: connecting with others who understand what you are going through.
- Healthy Relationships: Maintaining and improving relationships with family and friends.
- Avoiding alcohol and drugs: These can worsen depression and complicate treatment.
- Setting realistic goals: Break tasks into smaller steps and set priorities.

1.11 Ways in which social media can contribute to Depression:

1. **Social Comparison**:

- Idealized Images: Frequently, social media platforms are used to display idealized and curated pictures of people's lives, thereby leading their users to make unfavourable comparisons with one another and consequently feel inadequate, have low self-esteem, and be dissatisfied.
- Highlight Reels: The achievements and happy moments of those who use these sites are always being posted, making others feel like their lives are not as fulfilling or successful.

2. **Cyberbullying and Online Harassment**:

- Negative Interactions: Cyberbullying thrives well on such platforms, as they are often used to pass negative comments that may have a lasting impact on one's mental health when subjected to such harassment.
- Persistent Bullying: Cyberbullying has no time limit since it can happen at any hour of the day or night, thus making it hard for victims to run away from their abusers, unlike in traditional forms where bullying stops when school ends, for example.

3. **Addiction and Compulsive Use**:

   - Extreme consumption: Overspending too much time on social media may cause people to neglect genuine relationships and activities, leading them to be alone.
   - Disruption of Sleep: The usage of social platforms late into the night can disrupt sleeping patterns, which causes tiredness and worsens depression symptoms.

4. **Exposure to Negative Content:**

   - Disturbing News: A continuous viewing of adverse news like crimes and disasters, among others, can make one have a negative attitude towards life, thereby feeling powerless.
   - Toxic Communities: Interacting within groups that propagate bad behavioural patterns or mindsets will solidify wrong convictions, hence intensifying depression.

5. **Impact on Self-Image and Body Image**:

   - Unrealistic Standards: Being exposed to pictures and articles that advertise unrealistic beauty standards can detrimentally affect self-respect for physical appearance and body image perception.
   - Body Shaming: The deposition of discouraging remarks concerning the outlook can lead to dissatisfaction with one's body and eventual mental illness due to depression.

1.12 How can Sentiment Analysis help in detecting Depression from social media Texts

1. **Identifying Negative Sentiment**:

   - Depression is an illness that often involves feeling down all the time. Sentimental evaluation can be used to recognize:
   - Negative Words and Phrases: When certain words such as sadness, hopelessness, and other types of negative feelings are frequently utilized.
   - Emotional Polarity: A depressed mood may be indicated by an overall negative score for many posts on multiple platforms.

2. **Contextual Analysis:**

    - Contextual Usage: Differentiating between standard usage and signs of depression in language (for example, “I feel so empty” versus “the empty room”) through knowledge of where keywords appear together within sentences.
    - Phrase-Level Sentiment: Analysing sentiment at the phrase or sentence level to capture nuanced expressions of distress.
    - Depressive traits show up in posts on social networks. It is feasible that, through sentiment analysis, we can establish if:
    - Self-Worthlessness and Guiltiness: This can be indicated by phrases showing self-blame or low self-esteem.
    - Loss of Interest: This can be shown by phrases indicating no longer finding interest in activities previously enjoyed.
    - Hopelessness and Helplessness: Hopelessness is an expression of despair, while helplessness refers to feeling unable to cope with a situation.

3. **Machine Learning Models**: Using Machine Learning with Sentiment Analysis can Increase the Ability to Detect:

    - Training on Depressive Language: Models are trained on datasets of depressed people’s social media posts to learn how to pick out even the smallest cues.
    - Feature Extraction: Identifying important language traits that are associated with depression, like frequent use of negative adjectives and first-person pronouns.

4. **Multimodal Analysis:** Combining sentiment analysis with other data sources:

    - Text and Metadata: When textual information is evaluated together with metadata such as posting frequency, times of posts, and patterns of interaction.
    - Image and Video Analysis: This involves looking at visual content alongside written words on platforms that allow for multimedia sharing to gain deeper insight into one’s emotional well-being.

5. **Early Detection:** Depression can be detected early through sentiment analysis.

- Proactive Monitoring: This entails regular checks on social media updates, which may reveal signs of depressive feelings before they become severe enough to cause harm, thus enabling timely intervention.
- Alerts and Notifications: alerting mental health professionals or trusted individuals when an individual shows signs of being extremely depressed.

2. LITERATURE REVIEW:-

The proposed model is superior to the Linear SVM in predicting depression. It offers improved accuracy, precision, recall, and F1-score compared to the Linear SVM.The accuracy achieved in predicting depression using Multinomial Naive Bayes and Logistic Regression is 98%.A Deep Learning ensemble model is suggested for sentiment analysis, demonstrating better performance than existing models. There is potential for sentiment analysis on various social media platforms, including during the COVID-19 pandemic. Machine Learning techniques are effective in detecting signs of depression in social media texts. Some challenges include imbalanced data and reliance on labeled data. The importance of sentiment analysis in monitoring consumer opinions and business intelligence cannot be overlooked. Naive Bayes and SVM have been identified as powerful tools for stakeholders to manage brand perception based on expressed opinions

2.1 LITERATURE SURVEY:

S.NO /Link	Title	Authors	Abstract	Tools	Algorithm	Evaluation Metrics	Advantages	Disadvantages	Summary
1	Depression detection from social network data using machine learning techniques	Md. Rafqul Islam,Ashir Ahmed,Abu Raihan M. Kamal,Hua Wang,Anwaar Ulhaq	Social networks have been developed as a great point for its users to communicate with their interested friends and share their opinions, photos, and videos refecting their moods, feelings and sentiments.	Python,MATLAB	KNN,SVM,Decision Tree	evaluation metrics such as accuracy, precision, recall, and F1-score were commonly used to assess model performance. Detecting depression from social media text is a valuable application of machine learning, providing insights into mental health among social network users.	1.Early Intervention 2.Scalability 3.Objective Assessment 4.Privacy Preservation:	1.Ethical Concerns 2.Data Quality and Noise 3.Contextual Challenges 4.Bias and Generalization	In this paper we have exhibited the capability of using Facebook as a tool for measuring and detecting major depression among its users.
2	Survey on sentiment analysis: evolution of research methods	Jingfeng Cui, Zhaoxia Wang,Seng-Beng Ho1,Erik Cambria4	Many literature reviews on sentiment analysis involving techniques, methods, and applications have been produced using diferent survey methodologies and tools. but there has not been a survey dedicated to the evolution of research methods and top-ics of sentiment analysis.	Sentiment analysis , Keyword co-occurrence analysis , Evolution analysis · Research methods , Research topics	Sentimental Analysis	sentiment analysis finds applications across diverse fields, including business, finance, politics, education, and services. Researchers, governments, institutions, and companies alike recognize its significance	1.Early Intervention and Decision-Making 2.Scalability and Efficiency 3.Objective Insights 4.Business Application	1.Ethical Concerns 2.Data Quality and Noise 3.Contextual Challenges 4.Bias and Generalization	The survey explores the evolution of research methods and topics in sentiment analysis.It uses keyword co-occurrence analysis and a community detection algorithm.The study uncovers trends, hotspots, and practical insights for researchers.
3	Unveiling the Power of Social Media Analytics	Weiguo Fan,Michael D. Gordon	To harness the potential of UGCs, organizations must	Python,Jupyter	Sentimental Analysis	Measures the number of engagements	Trending Analysi	track social media	The authors would like to thank Jonathan Woody at

			develop the capability to collect, store, and analyze social media data. This process aims to extract actionable knowledge for decision-making and forecasting. As a consequence of these developments, social media analytics has emerged as a vital area of study.			(reactions, comments, and shares) your content receives as a percentage of your audience. Helps gauge audience interest and signals social media algorithms to expand your reach.	s	return on investment	Virginia Tech for his data collection support related to various part of this article. We would also like to thank the associate editor and anonymous reviewers for their valuable comments in helping us improve the quality of the paper.
4	A textual-based featuring approach for depression detection using machine learning classifiers and social media texts	Raymond Chionga,Gregorius Satia Budhia,Sandeep Dhakala,Fabian Chiongc	Depression is one of the leading causes of suicide worldwide. However, a large percentage of cases of depression go undiagnosed and, thus, untreated. Previous studies have found that messages posted by individuals with major depressive disorder on social media platforms can be analysed to predict if they are suffering, or likely to suffer, from depression.	Python,Jyputer,	KNN,SVM,Decision Tree,Random Forest	The study proposes a textual-based featuring approach for depression detection. It focuses on analyzing social media texts (such as Twitter, Facebook, Reddit, and electronic diaries) to identify signs of depression. Importantly, this approach doesn't rely on explicit keywords related to depression.	1.Non-Invasive 2.Scalability 3.Early Detection	1.False Positives 2.Privacy Concerns 3.Contextual Challenges	this research provides a promising avenue for leveraging machine learning and social media data to improve depression detection. However, it also highlights the need for careful consideration of privacy and context
5	Whatsapp Chat Analyzer	Ravishankara K, Dhanush, Vaisakh, Srajan	The most used and efficient method of communication in recent times is an application called WhatsApp. WhatsApp chats consists of various kinds of conversations held	Python,Json,DART, Flutter,R	Sentimental Analysis	Understand the overall chat volume.Measure the extent of communication. Identify multimedia content.	1.Non-Invasive 2.Scalable 3.Early Insights 4.Interactive	1.False Positives 2.Privacy Concerns 3.Contextual Challenges	WhatsApp Chat Analysis empowers users to gain actionable insights from their chat history. It's a valuable tool for businesses and individuals alike, but careful

			among group of people. This chat consists of various topics. This information can provide lots of data for latest technologies such as machine learning.			Visualize chat activity over time.	Reports		consideration of privacy and context is essential.
6	Chat analysis on whatapp using machine learning	Dr.T.Siva Ratna Sai,T.Naga Nandini2,M.Harsha Vardhan,B.Sivaji, R.Sai Kalyan	The most used and efficient method of communication in recent times is an application called WhatsApp. WhatsApp chat analyzer is the application deployed on Heroku web which provide analysis of WhatsApp group. This is the combination of machine learning and NLP.	Python,R ,Pytroch	Sentimental Analysis	Measures overall correctness of predictions. Proportion of true positive predictions among all positive predictions.Proportion of true positive predictions among actual positive instances.	1.Non-Invasive 2.Scalability 3.Early Insights 4.Interactive Reports	1.False Positives 2,Privacy Concerns 3.Contextual Challenges	The system was written in Python, and NumPy, Pandas, Matplotlib, and Seaborn were among the Python libraries used. The analysis was able to illustrate the amount of participation of the various people on the specified WhatsApp group at the conclusion of the task, and the findings were as anticipated.
7	WhatsApp Chat Sentiment Analyzer	Aditya Jadhav,Suraj Patil,Mujtaba Shaikh	WhatsApp chat analyzer is a web application which was developed to provide analysis of WhatsApp group chats. Python libraries like matplotlib, re, seaborn, streamlit, pandas and some conceptual knowledge of NLTK are in use here. This is the combination of NLP, NLTK and machine learning.	Python,NLP	LSTM,BERT	Measures overall correctness of predictions. Proportion of true positive predictions among all positive predictions.Proportion of true positive predictions among actual positive instances.	1.Automated Insights 2.Scalability 3.Early Detection 4.Business Application	1.False Positives 2.Privacy Concerns 3.Data Quality 4.Resource-Intensive	WhatsApp Chat Sentiment Analysis is a powerful tool for understanding user emotions. Choose the right model based on your dataset size, resources, and accuracy requirements. Always consider privacy and ethical implications

8	WhatsApp Network Group Chat Analysis Using Python Programming	Blessing Nwamaka Iduh	WhatsApp is an instant messaging application which enables users to send and receive messages in real time. It is a platform that has created an enabling environment for users to communicate with friends, groups and business partners at a cost of only a little internet access.	Python,Pandas,Numpy,Matplot,seaborn	Sentimental Analysis	Measures overall correctness of predictions. Proportion of true positive predictions among all positive predictions.Proportion of true positive predictions among actual positive instances.	Gain insights into group dynamics, active participants,peak messaging times.	1.Privacy concerns 2.Limited context	WhatsApp group chat analysis is a fascinating project that combines data science and social interactions. By building your own WhatsApp analyzer, you'll uncover hidden patterns and enhance your understanding of group dynamics.
9	Community converzation analyzer	Tanmay Gupta,Samarth Singh	The tool utilizes machine learning and natural language processing techniques, along with Python libraries such as Pandas, Matplotlib, and Seaborn, to analyse chat data and generate various visualizations.	Python,Seaborn,NLP,Pandas,Numpy	Sentimental Analysis	Assess the quality of community conversations using appropriate metrics.Consider both quantitative and qualitative aspects.	1.Rich Insights 2.Contextual Understanding 3.Actionable Recommendation	1.Privacy Concern 2.Subjectivity 3.Resource-Intensive	Community Conversation Analysis bridges theory and practice.By unraveling conversational patterns, we enhance community well-being and foster meaningful connections.
10	A Comprehensive data analyisis on fudma as whatapp group chat	Abubakar Ahmad,Nuhu Abdulhafiz A.	WhatsApp is an instant messaging application for information exchange in real time. It is a medium for communication and interaction among individuals, groups, institutions and business partners.	Python,Data Mining	Sentimental Analysis	The study employed data mining techniques to analyze the chat content.Metrics such as the number of messages, frequency of interactions, and participatio	WhatsApp groups facilitate instant communication and interaction among individuals, groups, institutions, and busine	whatsApp groups can become overwhelming due to the sheer volume of messages.	The research provides insights into the functioning of the FAM WhatsApp group, emphasizing the need for effective communication channels in academic settings

						n patterns were examined.	ss partners		
11	A review on sentiment analysis and emotion detection from text	Pansy Nandwani,Rupali Verma	Social networking platforms have become an essential means for communicating feelings to the entire world due to rapid expansion in the Internet era. Several people use textual content, pictures, audio, and video to express their feelings or viewpoints.	Afective computing,Natural language processing,Opinion mining,Pre-processing,Word embedding	LSTM,Sentimental Analysis,Navie Bayes,Decision Tree	Accuracy, precision, recall, F1-score, and AUC-ROC. Similar to sentiment analysis metrics.	provides deeper insights into users' emotional states.	Ambiguity in emotional expressions.	sentiment analysis helps us understand the overall sentiment (positive, negative, or neutral) in text, while emotion detection dives deeper into specific emotional states. Both fields face challenges related to context, data quality, and model performance.
12	Sentiment Analysis in Social Media and Its Application: Systematic Literature Review	Zulfadzli Drus,Haliyana Khalid	The data can be converted into valuable information by using sentiment analysis. A systematic review of studies published between 2014 to 2019 was undertaken using the following trusted and credible database including ACM, Emerald Insight, IEEE Xplore, Science Direct and Scopus.	Machine Learning, Lexical-based	Sentimental Analysis	Overall correctness of predictions. Proportion of true positive predictions among all positive predictions.	Provides insights into user opinions, brand perception, and public sentiment. Informs decision-making for businesses and organizations.	Struggles with sarcasm, irony, and context-dependent sentiments. May misinterpret nuanced expressions.	Sentiment analysis in social media remains a dynamic field, constantly evolving to tackle real-world challenges. As we navigate the vast sea of unstructured textual data, refining sentiment analysis techniques remains an exciting pursuit.

13	Sentiment Analysis and Text Classification for Social Media Contents Using Machine Learning Techniques	Aditya Bhardwaj	Automated text classification has been considered a vital approach to manage and process unstructured documents generated from social media and plain text in digita l forms. Machine learning techniques are found to be worthy of text mining.	Text Mining, Social Media, Sentiment Analysis, Feature Selection , Machine Learning	Naive Bayes Support Vector Machines (SVM) Decision Trees Random Forest AdaBoost Gradient Boosting	Sentiment Analysis (SA) called opinion mining also or polarity mining deals with computational linguistics,natural language processing, and other te xt analytics methods.	1.Insights 2.Decision-Making	1.Context Challenges 2.Nuanced Expressions 3.Multilingual Data	sentiment analysis empowers businesses to navigate the digital landscape by deciphering the collective voice of social media users
14	NLP and Sentiment Analysis: The Good, The Bad, and In-between	Ali Jan	The study's first part is dedicated to advancing a theoretical contribution by understanding various methods' similarities, methodologies, findings, and challenges.	NLP,Python	sentimental Analysis,Logistic Regression,Decision Tree Classifier,Gradient Boasting Classifier	1.Accuracy 2.Precision and Recall 3.F1 Score	1.Efficiency 2.Decision Support	1.Contextual Limitations 2.Customization Needed	It can also facilitate researchers in understanding the nature of the work and its intensity primarily related to COVID-19 sentiment analysis in the larger research community.
15	Twitter Sentiment Analysis The good, the bad and the neutral!	Ayushi Dalmia,Manish Gupta,Vasudeva Varma	Sentiment Analysis in Twitter. We participated in Sub-task B: Message Polarity Classification. The task is a message level classification of tweets into positive, negative and neutral sentiments.	Python	Classification	F1 Score: On the test data, their system achieved an F1 score of 59.83%. On the progress data, the F1 score improved to 67.04%.	1.Real-Time Insights 2.Business Intelligence 3.Political Campaigns 4.Social Research	1.Ambiguity 2.Informal Language 3.Context Dependency	Twitter sentiment analysis is a dynamic field with practical applications across domains. While challenges exist, innovative techniques continue to evolve, enabling us to uncover the emotional landscape of the Twitterverse.
16	Improving Sentiment Analysis for Social Media Applications Using an Ensemble Deep Learning Language Model	Ahmed Alsayat	Finding the most suitable classification algorithms for this kind of data is challenging. Within this context, models of deep learning for	Machine learning, Deep learning	Sentiment analysis,Data mining,Ensemble algorithms,CNN,LSTM	The researchers conducted several experiments using their own Twitter coronavirus hashtag	1.Real-Time Insights 2.Business Intelligence 3.Rese	1.Ambiguity 2.Context Dependency	Sentiment analysis using ensemble deep learning models opens exciting possibilities for understanding the emotional landscape of social media. As data

			sentiment analysis can introduce detailed representation capabilities and enhanced performance compared to existing feature-based techniques.			dataset and publicly available review datasets from Amazon and Yelp. Statistical analyses demonstrated that their proposed models outperformed other approaches in terms of classification accuracy.	arch and Monitoring		continue to grow, innovative techniques will play a crucial role in decoding sentiments across diverse platforms
17	Sentiment Analysis in Social Media Data for Depression Detection Using Artifcial Intelligence: A Review	Nirmal Varghese Babu,E.race Mary Kanaga	Sentiment analysis is an emerging trend nowadays to understand people's sentiments in multiple situations in their quotidian life. Social media data would be utilized for the entire process ie the analysis and classifcation processes and it consists of text data and emoticons, emojis, etc.	Machine Learning, Deep learning, NLP	Classification,Nltk	Previous studies have explored both binary and ternary classification, but multi-class classification provides more precise results.	1.Rich Data Source 2.Real-Time Insights 3.Potential for Early Detection	1.Noise and Ambiguity: 2.Privacy Concerns: 3.Model Complexity	Sentiment analysis using social media data is a powerful tool for detecting apprehensiveness or dejection. Leveraging artificial intelligence techniques, especially deep learning algorithms, enhances precision.
18	Predicting Anxiety, Depression and Stress in Modern Life using Machine Learning Algorithms	Anu Priyaa,Shruti Garga,Neha Prerna Tiggaa	these algorithms, data were collected from employed and unemployed individuals across different cultures and communities through the Depression, Anxiety and Stress Scale questionnaire (DASS 21).	Machine Learning ,Python	KNN,CNN,Navie bayes,SVM	Common evaluation metrics include precision, recall, F1-score, and specificity. The F1-score helps balance precision and recall, especially when dealing	Machine learning models can handle large amounts of data efficiently.	1.Interpretability 2.Data Quality 3.Overfitting	Machine learning algorithms offer promising avenues for predicting anxiety, depression, and stress.Researchers continue to refine and enhance these models to improve their accuracy and applicability.

			Anxiety, depression and stress were predicted as occurring on five levels			with imbalanced classes.			
19	Sentiment Analysis for Depression based on Social Media Stream	A. Punithn,K.Kiruthigaipriya,B.Vanitha,U.S.Harshanaa 4	- This project deals with depressive disorder. In the existing system challenging to process the unusual symbol, stop words and punctuation and major drawback of this system is aid of human (i.e. person need to extract and feed the inputs to the system). The time consumption of this system is more, the intimation of messages are not done at early stage of depressive disorder.	Python,TCP/IP protocal	sentimental analysis,LSTM,BLSTM,GRU	Precision, recall, F1-score, and specificity are commonly used evaluation metrics. F1-score balances precision and recall, especially when dealing with imbalanced classes.	1.Efficiency 2.Complex Patterns 3..Real-time Monitoring	1.Interpretability 2.Data Quality 3.Overfitting:	Sentiment analysis on social media data provides valuable insights into users' emotional states. Multi-class classification using deep learning algorithms yields higher precision during sentiment analysis. Researchers continue to refine these techniques to enhance their accuracy and applicability
20	Sentiment Analysis from Bengali Depression Dataset using Machine Learning	Md. Rafidul Hasan Khan	Sentiment Analysis is one of the advanced matters of natural language processing. Sentiment analysis determines a particular pole of a paragraph. Our purpose is to find the sentiment from the Bengali paragraph which is happy or sad using various types of machine learning	Machine Learning ,Python, Deep Learning	NLP,sentimental analysis,Depression Detection	About Accurary	Machine learning models efficiently handle large amounts of unstructured data.	1.Interpretability 2.Data Quality	Sentiment analysis in Bengali text provides insights into emotional states. Researchers continue to refine techniques to enhance accuracy and applicability.

			classification analysis algorithms.						
21	The Power of Social Media Analytics	Weiguo Fan,Michael D. Gordon	In the early days of social media, PR agencies would monitor customersȜ posts on a businessȜs own website in an attempt to identify and manage disgruntled customers. With the explosion in the number of social media sites and the volume of use on them, this is not nearly enough.	Machine Learning	sentimental Analysis	Engagement Metrics,Accurary	1.Real-time Insights 2.Audience Segmentation 3.Competitor Analysis 4.Campaign Optimization	1.Privacy Concerns 2.Data Noise 3.Algorithm Bias 4.Interpretability	Social media analytics empowers businesses, marketers, and researchers. By harnessing data-driven insights, organizations can enhance engagement, improve brand perception, and drive growth.
22	Deep Learning-Based Depression Detection from Social Media: Comparative Evaluation of ML and Transformer Techniques	Biodoumoye George Bokolo,Qingzhong Liu	Abstract: Detecting depression from user-generated content on social media platforms has garnered significant attention due to its potential for the early identification and monitoring of mental health issues. This paper presents a comprehensive approach for depression detection from user tweets using machine learning techniques.	Machien Learning, Python	NLP,Sentimental Analysis,DeepLearning models	The study assesses model performance using metrics such as accuracy, precision, recall, and F1 score.	1.Advanced Transformer-Based Models 2.Early Detection Potential	N/A	This research contributes to the growing field of mental health analysis based on user-generated content. By harnessing machine learning techniques and large language models, we can enhance our ability to detect and address mental health concerns through social media data.

23	Sentiment Analysis in Social Media Data for Depression Detection Using Artifcial Intelligence: A Review	Nirmal Varghese Babu,E.Grace Mary Kanaga	Sentiment analysis is an emerging trend nowadays to understand people's sentiments in multiple situations in their quotidian life. Social media data would be utilized for the entire process ie the analysis and classifcation processes and it consists of text data and emoticons, emojis, etc. Many experiments were conducted in the antecedent studies utilizing Binary and Ternary Classifcation whereas Multi-class Classifcation gives more precise and precise Classifcation.	Deep Learning, Machine Learning	Sentimental analysis,NLP	The study utilizes social media data containing texts, emoticons, and emojis.	1.Advanced Models 2.Early Detection	N/A	This research contributes to the field of mental health analysis based on user-generated content. By leveraging artificial intelligence techniques, we can improve depression detection using social media data.
24	Automatic detection of depression symptoms in twitter using multimodal analysis	Ramin Safa,Peyman Bayat,Leila Moghtader	Depression is the most prevalent mental disorder that can lead to suicide. Due to the tendency of people to share their thoughts on social platforms, social data contain valuable information that can be used to identify user's psychological states. In this paper, we provide an automated approach to collect and evaluate tweets based on self-	NLP,Python	Classification,Feature Extraction,	Precision, recall, F1-score, and specificity are commonly used evaluation metrics. F1-score balances precision and recall, especially when dealing with imbalanced classes.	Textual Features	N/A	The study contributes to mental health analysis based on user-generated content. By combining textual and visual cues, we can better understand and address depression symptoms in online communities.

			reported statements and present a novel multimodal framework to predict depression symptoms from user profles.						
25	Text-Based Detection of the Risk of Depression	Jana M. Havigerová,Jirí Haviger,Dalibor Kucera,Petra Hoffmannova	This study examines the relationship between language use and psychological characteristics of the communicator. The aim of the study was to find models predicting the depressivity of the writer based on the computational linguistic markers of his/her written text. Respondents' linguistic fingerprints were traced in four texts of different genres. Depressivity was measured using the Depression, Anxiety and Stress Scale (DASS-21).	Machine Learning	Classification	Accuracy and classification error rate are commonly used evaluation metrics. Psycholinguistic features play a crucial role in improving model performance.	Research,findings	N/A	The leading motivation for our research is to find ways to use automatic analysis of texts (such as cover letters, letters from holidays, blogs, and comments on social networks) to create predictive models that will reliably detect individuals at risk of a mental disorder (such as depression in the present study) so that they can be provided with help as early as possible.

26	Depression prognosis using natural language processing and machine learning from social media status	Md. Tazmim Hossain, Md. Arafat Rahman Talukder, Nusrat Jahan	Depression is an acute problem throughout the world. Due to worst and prolong depression many people dies in every year. The problem is that most of the people are not concern of the fact that they are suffering from depression.	Machine Learning	NLTK,NLP,Classification	Precision, recall, F1-score, and specificity are commonly used evaluation metrics. F1-score balances precision and recall, especially when dealing with imbalanced classes.	1.Efficiency 2.Complex Patterns 3..Real-time Monitoring	1.Ambiguity 2.Context Dependency	Our aim was to predict depression by using different social media posts. Especially, people in poor and least developed countries are suffering from this disease. Therefore, they are suffering from depression for a long time which is reducing their ability to survive and it slowly pushing them towards death.
27	Emotions from text: machine learning for text-based emotion prediction	Cecilia Ovesdotter Alm,Dan Roth,Richard Sproat	Text contains attitudinal, and more specifically,emotional content. This paper explores the text-based emotion prediction problem empirically, using supervised machine learning with the SNoW learning architecture. The goal is to classify the emotional affinity of sentences in the narrative domain of children's fairy tales, for subsequent usage in appropriate expressive rendering of text-to-speech synthesis.	Machine Learning	NLP,NLTK,Feature Extraction	Accuracy and classification error rate are commonly used evaluation metrics. Psycholinguistic features play a crucial role in improving model performance.	emotion prediction task,text-to-speech	N/A	This paper has discussed an empirical study of the text-based emotion prediction problem in the domain of children's fairy tales, with child-directed expressive text-to speech synthesis as goal. Besides reporting on encouraging results in a first set of computational experiments using supervised machine learning, we have set forth a research agenda for tackling the TEP problem more comprehensively.
28	Analysis of Social Media Sentiment for Depression Prediction using Supervised Learning and	Yogesh Sahu,Dr.Pinaki Ghosh	Sentiment analysis is a new trend in understanding people's emotions in a variety of scenarios in their	Machine Learning, Python	Sentimental Analysis,Radial Basis Function,Classification	Accuracy,Performance	prediction accuracy as compare than	N/A	Hence, depression prediction in software is better predict through proposed method SL-RBF

	Radial Basis Function		daily lives. Social media data, which includes text data as well as emoticons, emojis, and other images, would be used throughout the process, including the analysis and categorization procedures.				Linear SVM.		(Supervised Learning with Radial Basis Function).
29	Sentiment analysis for depression detection	Swarada Jalukar , Arati Ratnaparkhi , Priyanka Shinde , Simran Kunkulol , Vinaya Kulkarni	population. Nowadays risk of early death is increasing due to mental illness which is mostly caused due to depression. Depression creates suicidal thoughts causing serious impairments in daily life. Sentiment analysis is a hot topic that's been on research for decades,which intends to find the nature of text and classifies into positive, negative and neutral.	Machine Learning ,SQL database	Naive Bayes,CNN, OpenCV	Performance on the Naive Bayes	Easy for detecting the depression	InAccuracy	with the help of Naïve bayes classifier for textual detection, google speech to text API and OpenCV library along with CNN will detect the depression level followed by suggestions of instant actions to be taken by patient.
30	Big data analytics on social networks for real-time depression detection	Jitimon Angskun,Suda Tipprasert,Thara Angskun	Several depression suferers disclose their actual feeling via social media. Thus, big data analytics on social networks for real-time depression detection is proposed. This research work detected the depression by analyzing both demographic characteristics	NLTK,Machine Learning, Deep Learning	SVM(RBF kernel),DT,Naive Bayes,Random Forest, FNN	Accuracy,Precision,Recall	Analysis the depression on the data	It have some Limit for access the data	Firstly, the number of participants could be increased for higher accuracy. Secondly, other features such as length of time spent on social networks and the number of likes on each post could be investigated for model construction. Tirdly, improving the model construction process using other

			and opinions of Twitter users during a two-month period after having answered the Patient Health Questionnaire-9 used as an outcome measure.						data such as emoticons and images could be performed. Finally, data could be gathered from multiple social networks.

Table 2.1: Literature Survey

2.2 GAP ANALYSIS:

- Recognition of depression in texts from social media by current systems is quite a challenge. This is because models often miss out on the informal and nuanced language used in these spaces resulting in either false negatives or false positives.
- Datasets for training depression recognition models are imbalanced in terms of class distribution. There are very few examples of depressive texts compared to non-depressive ones which makes it difficult for the model to learn and generalize well.
- Sometimes a model may fail to capture the meaning of words in a sentence hence incorrectly identifying a post as depressive. The reason behind this is that language used on social media is informal, diverse and contextual thus difficult for machines to comprehend.

2.3 OBJECTIVES:

- Create more complex models using LSTM networks that can understand and analyze informal language used in social media posts to improve detection accuracy.
- Apply methods for balancing classes in the training dataset. Use evaluation metrics accounting for class imbalance such as F1-score and precision-recall curves ensuring

balanced performance.

- Integrate contextual analysis methods into the model to better capture the meanings and tones of words or phrases used in tweets, comments etc., thereby making it easier to identify depressive contents accurately.

2.4 PROBLEM STATEMENT:

Creating a model to detect depression by analyzing social media texts through the application of a neural network. To improve the accuracy and balancing classes and while integrating contextual analysis methods.

3. DESGIN ANALYSIS:

3.1 MODEL ARCHITECTURE:

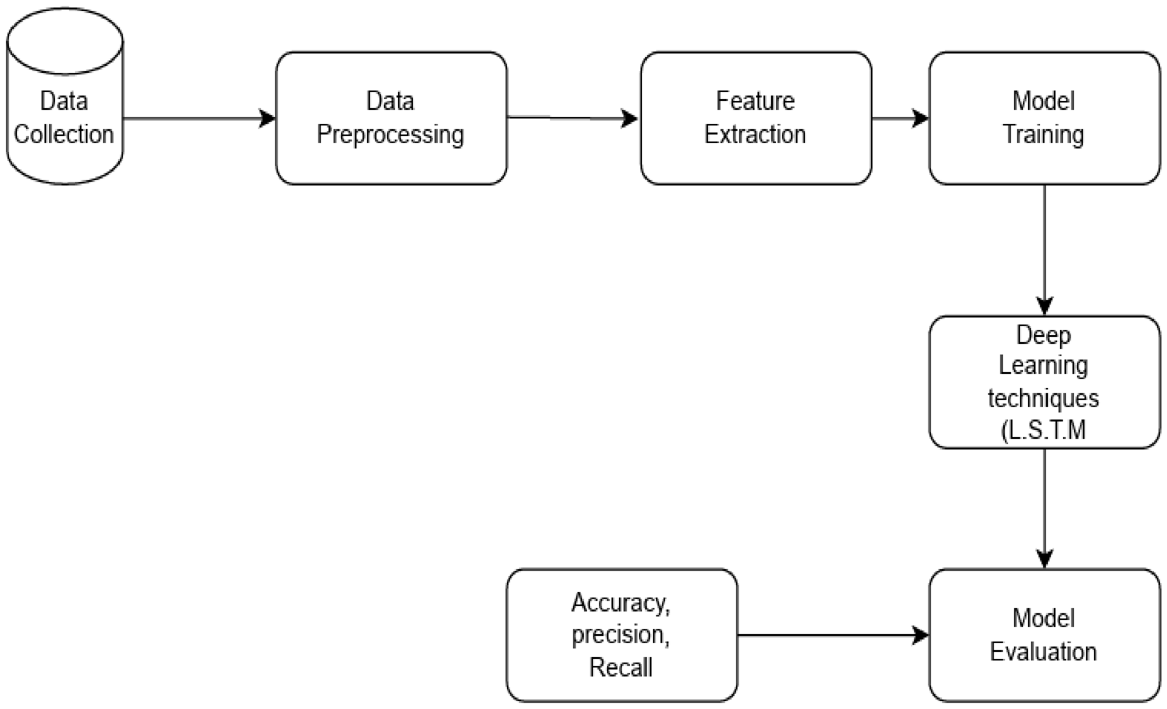


Fig 3.1: Model Architecture

3.1.1 Dataset Collection:

Data collection is the process of gathering and measuring information on variables of interest in an established, systematic fashion that enables one to answer stated research questions, test hypotheses, and evaluate outcomes. The data collection component of research is common to all fields of study, including physical and social sciences, humanities, business, etc. While methods vary by discipline, the emphasis on ensuring accurate and honest collection remains the same.

The goal of all data collection is to capture quality evidence that then translates to rich data analysis and allows the building of a convincing and credible answer to questions that have been posed. Regardless of the field of study or preference for defining data (quantitative or qualitative), accurate data collection is essential to maintaining the integrity of research. Both the selection of appropriate data collection instruments and the delineated instructions for their correct use reduce the likelihood of errors occurring.

3.1.2 Data Preprocessing:

Data pre-processing is the process of generating raw data for deep learning models. This is the first step in creating a machine-learning model. This is the most complex and time-consuming aspect of data science. Data pre-processing is required in deep learning algorithms to reduce their complexities. Data in the real world can have many problems. It can miss some elements or pieces of information. While incomplete or missing data is completely useless, adjusting and refining the data to make it valuable is the primary objective of data pre-processing.
It is an important step in the machine learning algorithm. Imagine a situation where you are working on an assignment at your college and the lecturer does not provide the raw headings and the idea of the topic. In this case, it will be very difficult for you to complete that assignment because raw data is not presented well to you. The same is the case with deep learning. Suppose the data pre-processing step is missing while implementing the algorithm. In that case, it will affect your work at the end, when it will be the final stage of applying the available data set to your algorithm.
While performing data pre-processing, it is important to ensure data accuracy so that it doesn't affect your deep learning algorithm at the final stage.

Pre-processing tasks involve:

- Remove links and images
- Lower Casing

- Remove hashtags
- Remove @ mentions
- Remove emojis
- Remove stop words.
- Remove punctuation
- Get rid of stuff like "what's" and make it "what is'
- Stem words so they are all in the same tense (e.g., ran -> run).

3.1.3 Tokenization:

Tokenization is a foundational step in the NLP pipeline that shapes the entire workflow. **It is** the process of dividing a text into smaller units known as tokens. **Tokens** are typically words or sub-words in the context of natural language processing. Tokenization is a critical step in many NLP tasks, including text processing, language modelling, and machine translation. The process involves splitting a string, or text, into a list of tokens. One can think of tokens as parts like a word is a token in a sentence, and a sentence is a token in a paragraph.

3.1.4 Stop Word Removal:

A stop word is a commonly used word (such as "the," "a," "an," or “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. The necessity of removing stop words in NLP is contingent upon the specific task at hand. For text classification tasks, where the objective is to categorize text into distinct groups, excluding stop words is common practice. This is done to channel more attention towards words that truly convey the essence of the text. As illustrated earlier, certain words like “there,” “book,” and “table” contribute significantly to the text’s meaning, unlike less informative words such as “is” and “on.” Conversely, for tasks like machine translation and text summarization, the removal of stop words is not recommended. In these scenarios, every word plays a pivotal role in preserving the original meaning of the content.

3.1.5 Stemming:

Stemming is one of several text normalization techniques that convert raw text data into a readable format for NLP tasks. It is a text pre-processing technique where the inflected form of a word is converted to one so-called “stem,” or root form, also known as a “lemma” in linguistics. [1] It is one of two primary methods, the other being lemmatization, which reduces inflectional variants within a text dataset to one morphological lexeme. In doing so, stemming aims to improve text processing in deep learning and information retrieval systems.

3.1.6 Lemmatization:

Lemmatization is a text pre-processing technique used in the natural language processing model that is used to break a word down to its root meaning to identify similarities. For example, a lemmatization algorithm would reduce the word better to its root word, or lemma, good. In both stemming and lemmatization, we try to reduce a given word to its root word. The root word is called a stem in the stemming process, and it's called a lemma in the lemmatization process.

3.1.7 Feature Extraction:

1. **Word2Vec:**

   Word2Vec is an approach for representing words as vectors used in natural language processing (NLP), where a word is converted into a continuous vector. Word2Vec models comprehend the meaning of words semantically, and this makes them applicable for different NLP exercises.

Key Concepts of Word2Vec:

1. **Embedding Vectors:**

Each word in a dictionary has a dimensional vector that represents it hugely. The concept behind these vectors is that words that carry the same meaning would also share these similar vectors, allowing for relevant comparisons as well as mathematical operations.

2. **Training Models:**

Word2Vec can be trained using two primary architectures, namely:
Continuous Bag of Words (CBOW): This architecture predicts a target word using context words, i.e., it predicts what word should come next given the surrounding words (in other terms).
Skip-Gram: It predicts the words around a specific word. It uses the central word to estimate the neighboring words.

3. **Dimensionality Reduction:**

One way to reduce the size of a vector space is via Word2Vec, which maps words into a continuous vector space where words that are similar in meaning have similar vectors. Normally, this is done in a lower-dimensional space, such as 100–300 dimensions.

4. **Context window:**

The number of surrounding words used in predicting the target word depends on the size of the context window. If the window size is bigger, it covers a wider area; if it is smaller, then only words close to each other can be considered.

2. **Sequence padding:**

To ensure that all sequences within a dataset have an equal length, sequence padding is a method used in natural language processing (NLP) and deep learning. This is very important for models such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), and transformers, which need input sequences of equal length to enable them to compute effectively.

The Way Sequence Padding Works:

1. **PaddingTokens:**
   Special tokens (mostly zero or specific padding tokens) are added to sequences so that they can be the same length as required.
2. **FixedLength:**
   The padding is often done based on the maximum sequence length in the dataset or some predetermined threshold, like the maximum length a model can handle.

3.1.8 Training:

- **Early Stopping:** Early Stopping is a regularization technique for deep neural networks that stops training when parameter updates no longer begin to yield improves on a validation set. In essence, we store and update the current best parameters during training, and when parameter updates no longer yield an improvement (after a set number of iterations) we stop training and use the last best parameters. It works as a regularizer by restricting the optimization procedure to a smaller volume of parameter space.
- **Epoch:** An epoch is when all the training data is used at once and is defined as the total number of iterations of all the training data in one cycle for training the machine learning model. Another way to define an epoch is the number of passes a training dataset takes around an algorithm. One pass is counted when the data set has done

both forward and backward passes. The number of epochs is considered a hyper parameter. It defines the number of times the entire data set has to be worked through the learning algorithm. Every sample in the training dataset has had a chance to update the internal model parameters once during an epoch. One or more batches make up an epoch. The batch gradient descent learning algorithm, for instance, is used to describe an Epoch that only contains one batch.

Learning algorithms take hundreds or thousands of epochs to minimize the error in the model to the greatest extent possible. The number of epochs may be as low as ten or high as 1000 and more. A learning curve can be plotted with the data on the number of times and the number of epochs. This is plotted with epochs along the x-axis as time and skill of the model on the y-axis. The plotted curve can provide insights into whether the given model is under-learned, over-learned, or a correct fit to the training dataset.

3.1.9 Evaluation:

Evaluation metrics can help you assess your model's performance, monitor your ML system in Production, and control your model to fit your business needs. Our goal is to create and select a model which gives high accuracy on out-of-sample data. It's Very crucial to use multiple evaluation metrics to evaluate your model because a model may Perform well using one measurement from one evaluation metric while may perform poorly

Using another measurement from another evaluation metric.

- **Accuracy**: The percentage of predictions that are correct.
- **Precisio*n***: The percentage of positive predictions that are actually positive.
- **Recall**: The percentage of actual positives that are predicted as positive.
- **F1 score**: A harmonic mean of precision and recall.

The choice of evaluation metric depends on the specific problem being solved. For example, if the goal is to minimize the number of false positives, then precision would be a good metric to use. If the goal is to minimize the number of false negatives, then recall would be a good metricto use.

It is important to note that no single evaluation metric can perfectly capture the performance of a machine learning model. It is often helpful to use multiple evaluation metrics to get a more complete picture of the model's performance.

4. MODULES AND IMPLEMENTATION

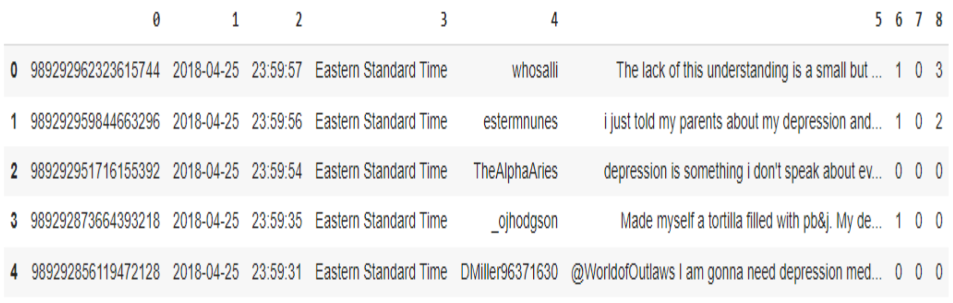
4.1 Data Collection:

Data collection is the process of gathering and measuring information on variables of interest, in an established systematic fashion that enables one to answer stated research questions, test hypotheses, and evaluate outcomes. The data collection component of research is common to all fields of study including physical and social sciences, humanities, business, etc. While methods vary by discipline, the emphasis on ensuring accurate and honest collection remains the same.

Two types of tweets were utilized in order to build the model: random tweets which do not necessarily indicate depression Sentiment140 data-set on Kaggle and depressed tweets which were extracted using the twitter API. Since there are no publicly available twitter data-set that indicates depression, the tweets were extracted according to the linguistic markers indicative of depression such as ”Hopeless”, ”Lonely”, ”Suicide”, ”Antidepressants”, ”Depressed”, etc. The random tweets dataset could be download from the kaggle website by the following link: https://www.kaggle.com/ywang311/twitter-sentiment/data.

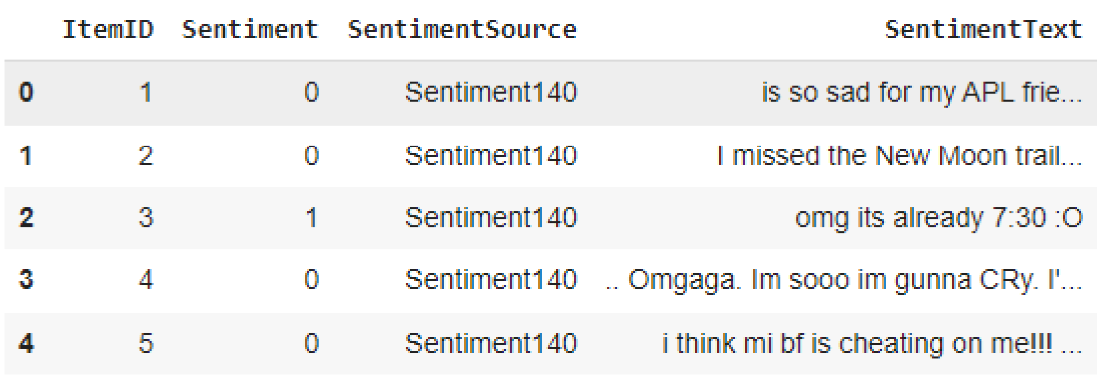
We also need the pretrained vectors for word2Vec model which provide by the google by using the following link: https://drive.google.com/uc?id=0B7XkCwpI5KDYNlNUTTlSS21pQmM&export=download
The goal for all data collection is to capture quality evidence that then translates to rich data analysis and allows the building of a convincing and credible answer to questions that have

been posed. Regardless of the field of study or preference for defining data (quantitative, qualitative), accurate data collection is essential to maintaining the integrity of research. Both the selection of appropriate data collection instruments and clearly delineated instructions for their correct use reduce the likelihood of errors occurring.



	0	1	2	3	4	5	6	7	8
0	989292962323615744	2018-04-25	23:59:57	Eastern Standard Time	whosalli	The lack of this understanding is a small but ...	1	0	3
1	989292959844663296	2018-04-25	23:59:56	Eastern Standard Time	estermnunes	i just told my parents about my depression and...	1	0	2
2	989292951716155392	2018-04-25	23:59:54	Eastern Standard Time	TheAlphaAries	depression is something i don't speak about ev...	0	0	0
3	989292873664393218	2018-04-25	23:59:35	Eastern Standard Time	_ojhodgson	Made myself a tortilla filled with pb&j. My de...	1	0	0
4	989292856119472128	2018-04-25	23:59:31	Eastern Standard Time	DMiller96371630	@WorldofOutlaws I am gonna need depression med...	0	0	0

Fig 4.1.1: Depressed Tweets

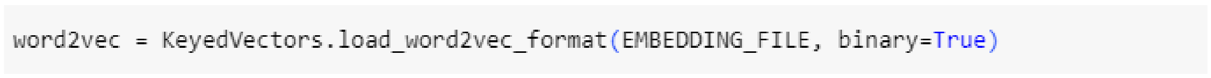


	ItemID	Sentiment	SentimentSource	SentimentText
0	1	0	Sentiment140	is so sad for my APL frie...
1	2	0	Sentiment140	I missed the New Moon trail...
2	3	1	Sentiment140	omg its already 7:30 :O
3	4	0	Sentiment140	.. Omgaga. Im sooo im gunna CRy. I'...
4	5	0	Sentiment140	i think mi bf is cheating on me!!! ...

Fig 4.1.2: Random Tweets

4.2 Pre-Processing:

Data pre-processing is the process of generating raw data for deep learning models. This is the first step in creating a machine-learning model. This is the most complex and time-consuming aspect of data science. Data pre-processing is required in deep learning algorithms to reduce its complexities. Data in the real world can have many problems. It can miss some elements or pieces of information. While incomplete or missing data is completely useless, adjusting and refining the data to make it valuable is the primary objective of data pre-processing.
The pretrained vectors for the Word2Vec model is from here. Using a Keyed Vectors file, we can get the embedding of any word by calling .word_vec(word) and we can get all the words in the model's vocabulary through Key_to_indexWord2Vec is used in the code because it is a powerful tool for representing words as numerical vectors. These vectors capture the semantic relationships between words, making them useful for a variety of natural language processing tasks.



```
word2vec = KeyedVectors.load_word2vec_format(EMBEDDING_FILE, binary=True)
```

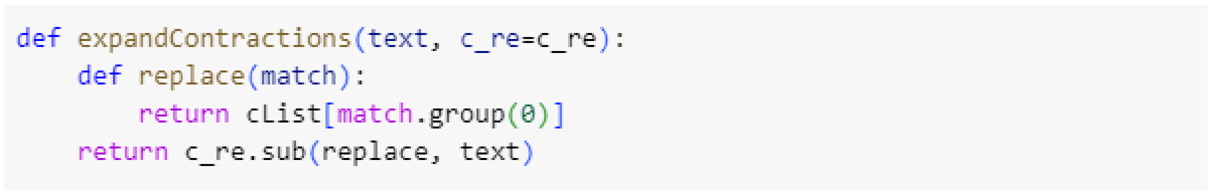
Fig 4.2.1: Word2vec

Raw text data might contain unwanted or unimportant text due to which our results might not give efficient accuracy, and might make it hard to understand and analyse. So, proper pre-processing must be done on raw data.

Pre-processing the tweets in order to:

- Remove links and images

- Remove hashtags
- Remove @ mentions
- Remove emoji's
- Remove stop words
- Remove punctuation
- Get rid of stuff like "what's" and making it "what is', STEM WORDS While performing data pre-processing, it is important to ensure data accuracy so that it doesn't affect your deep learning algorithm.



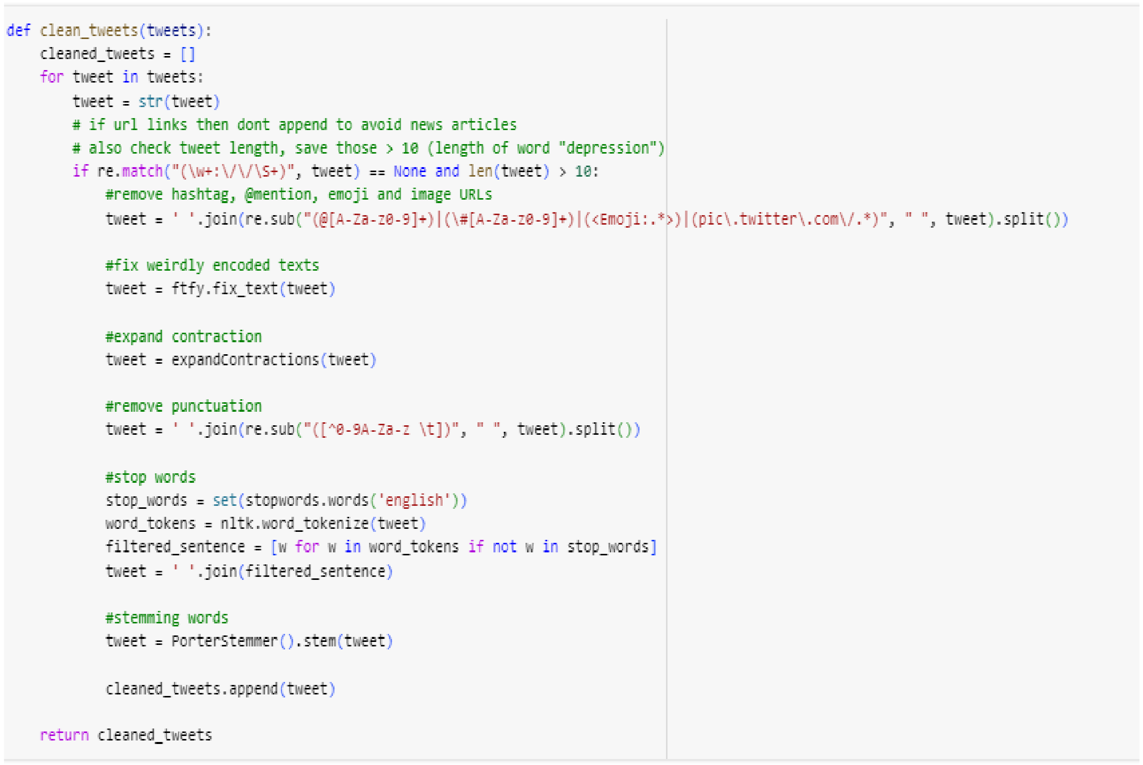
```
def expandContractions(text, c_re=c_re):
    def replace(match):
        return cList[match.group(0)]
    return c_re.sub(replace, text)
```

Fig 4.2.2: Expand Contractions Function

The expand Contractions function is used to replace contractions (shortened forms of words) with their full forms. This is useful for pre-processing text data, as it helps to standardize the text and make it easier for natural language processing models to understand.

4.3 Cleaning Dataset:

The clean tweets function takes a list of tweets as input and returns a list of cleaned tweets. This ensures that the cleaning process is case-insensitive. This uses a regular expression to remove any strings starting with http. This uses a regular expression to remove any strings starting with http. This uses a regular expression to remove any characters that are not alphanumeric or whitespace. This uses a regular expression to replace multiple consecutive spaces with a single space.



```python
def clean_tweets(tweets):
    cleaned_tweets = []
    for tweet in tweets:
        tweet = str(tweet)
        # if url links then dont append to avoid news articles
        # also check tweet length, save those > 10 (length of word "depression")
        if re.match("(\w+:\/\/\S+)", tweet) == None and len(tweet) > 10:
            #remove hashtag, @mention, emoji and image URLs
            tweet = ' '.join(re.sub("(@[A-Za-z0-9]+)|(\#[A-Za-z0-9]+)|(<Emoji:.*>)|(pic\.twitter\.com\/.*)", " ", tweet).split())

            #fix weirdly encoded texts
            tweet = ftfy.fix_text(tweet)

            #expand contraction
            tweet = expandContractions(tweet)

            #remove punctuation
            tweet = ' '.join(re.sub("([^0-9A-Za-z \t])", " ", tweet).split())

            #stop words
            stop_words = set(stopwords.words('english'))
            word_tokens = nltk.word_tokenize(tweet)
            filtered_sentence = [w for w in word_tokens if not w in stop_words]
            tweet = ' '.join(filtered_sentence)

            #stemming words
            tweet = PorterStemmer().stem(tweet)

            cleaned_tweets.append(tweet)

    return cleaned_tweets
```

Fig 4.3: Cleaning Dataset

4.4 Changing into Lower Case:

Lower casing is a common text pre-processing technique. The idea is to convert the input text into same casing format so that 'text', 'Text' and 'TEXT' are treated the same way. This is more helpful for text featurization techniques it helps to combine the same words together thereby reducing the duplication and get correct counts. In some use cases, like the tokenizer and vectorization processes, the lower casing is done beforehand.

Removal of Punctuations: One another common text pre-processing technique is to remove the punctuations from the text data. This is again a text standardization process that will help to treat 'hurray' and 'hurray!' in the same way. We also need to carefully choose the list of punctuations to exclude depending on the use case. For example, the string. Punctuation in python contains the following punctuation symbols!”#$%&\'()*+,-./:;<=>?@[\\]^_{|}~` .

Remove Image URLs: Find the indices of the start and end of each image URL. Remove the substring containing the URL from the original string.

Remove Emoji: Emoji’s are typically represented by a sequence of Unicode characters enclosed in angle brackets, e.g., <U+1F600>.Replace the angle brackets and Unicode characters of each emoji with an empty string.

Fix Weirdly Encoded Texts: Look for text that contains strange characters, such as mojibake or garbled symbols. This can be caused by incorrect encoding or decoding of the text. Weirdly encoded text occurs when the encoding of the text is not recognized or is incorrectly interpreted. By identifying the correct encoding and re-encoding the text, you can restore it to its original form.

Stop Word Removal: A Stop Word is a commonly used word (such as “the”, “a”, “an”, or “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. The necessity of removing stop words in NLP is contingent upon the specific task at hand. For text classification tasks, where the objective is to categorize text into distinct groups, excluding stop words is common practice. This is done to channel more attention towards words that truly convey the essence of the text. As illustrated earlier, certain words like “there,” “book,” and “table” contribute significantly to the text’s meaning, unlike less informative words such as “is” and “on. “Conversely, for tasks like machine translation and text summarization, the removal of stop words is not recommended. In these scenarios, every word plays a pivotal role in preserving the original meaning of the content.

Stemming: Stemming is one of several text normalization techniques that converts raw text data into a readable format for NLP tasks. It is a text pre-processing technique where the inflected form of a word to one so-called "stem," or root form, also known as a "lemma" in linguistics.1 It is one of two primary methods the other being lemmatization that reduces inflectional variants within a text dataset to one morphological lexeme. In doing so, stemming aims to improve text processing in deep learning and information retrieval systems.

Tokenization is a foundation step in NLP pipeline that shapes the entire workflow.

4.5 Tokenization:

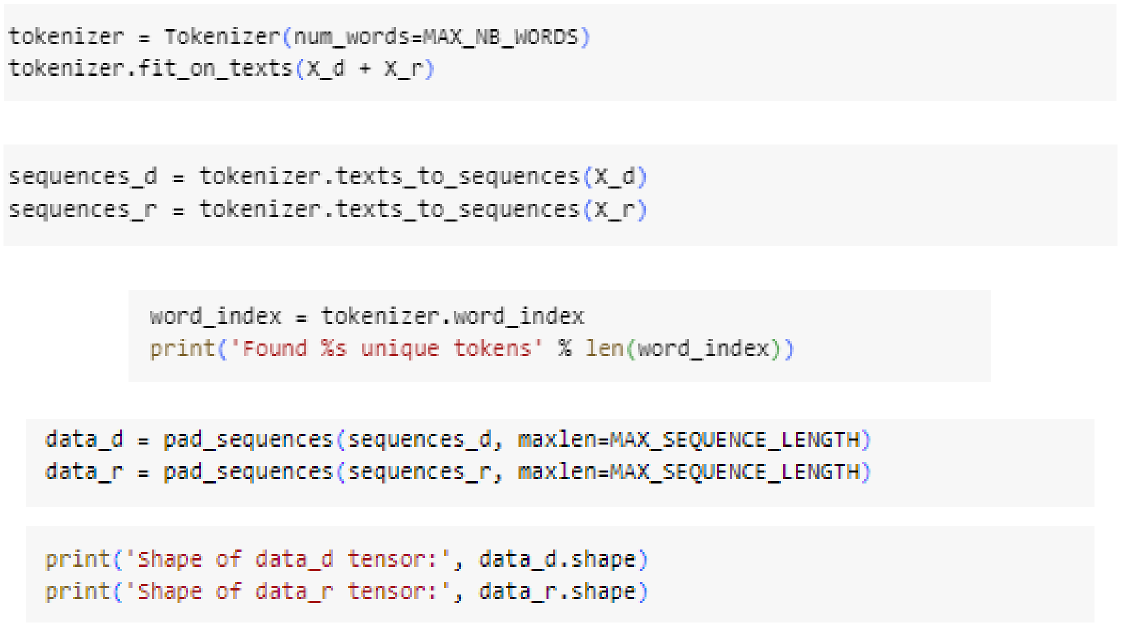
Tokenization is a foundation step in NLP pipeline that shapes the entire workflow. **It** is the process of dividing a text into smaller units known as tokens. **Tokens** are typically words or sub-words in the context of natural language processing. Tokenization is a critical step in many NLP tasks, including text processing, language modelling, and machine translation. The process involves splitting a string, or text into a list of tokens. One can think of tokens as parts like a word is a token in a sentence, and a sentence is a token in a paragraph.

The process of word tokenization involves identifying the boundaries between words, which can be achieved through various techniques. The most common approach is to use whitespace as a delimiter between words, although this can lead to issues with contractions, hyphenated words, or compound words.

NLTK or spaCy, which provides more sophisticated methods for tokenizing text, such as rule-based or statistical models. Word token removal can also be customized based on specific requirements or context. For instance, in sentiment analysis, emoticons or slang words may need to be removed to ensure accurate sentiment classification. In machine translation, specific words or phrases may need to be removed or replaced to account for differences in language structures or syntax. However, it is essential to carefully consider the potential drawbacks of word token removal, as it can lead to the loss of important information or context. For instance, removing specific words or phrases can impact the overall meaning or tone of the text, which can lead to incorrect or misleading results.

Using a Tokenizer to assign indices and filtering out unfrequently words. Tokenizer creates a map of every unique word and an assigned index to it. The parameter called num_words indicates that we only care about the top 20000 most frequent words.

Applying the tokenizer to depressive tweets and random tweets data.



```
tokenizer = Tokenizer(num_words=MAX_NB_WORDS)
tokenizer.fit_on_texts(X_d + X_r)
```

```
sequences_d = tokenizer.texts_to_sequences(X_d)
sequences_r = tokenizer.texts_to_sequences(X_r)
```

```
word_index = tokenizer.word_index
print('Found %s unique tokens' % len(word_index))
```

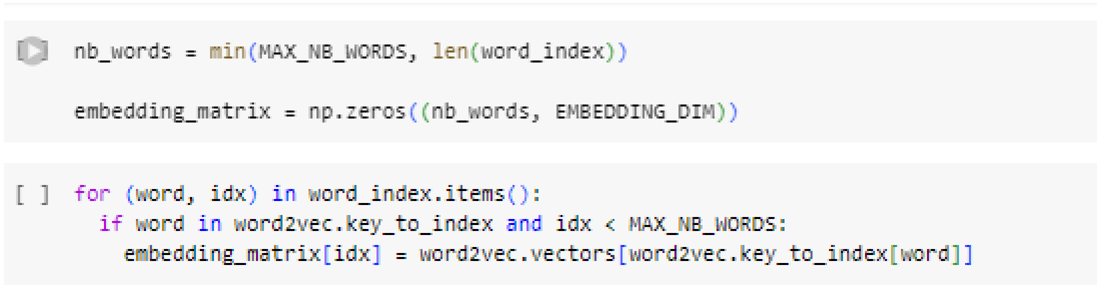
```
data_d = pad_sequences(sequences_d, maxlen=MAX_SEQUENCE_LENGTH)
data_r = pad_sequences(sequences_r, maxlen=MAX_SEQUENCE_LENGTH)
```

```
print('Shape of data_d tensor:', data_d.shape)
print('Shape of data_r tensor:', data_r.shape)
```

Fig 4.5: Tokenization

4.6 Embedding Matrix:

The embedding matrix is a rectangular matrix with dimensions n x m, where. n is the number of words in the vocabulary. m is the dimension of the embedding vectors. In this case, m = 300 and n = 20000. The value of n is determined by taking the minimum between the number of unique words in the tokenizer and the maximum number of words specified (max_words).



```
nb_words = min(MAX_NB_WORDS, len(word_index))

embedding_matrix = np.zeros((nb_words, EMBEDDING_DIM))
```

```
[ ] for (word, idx) in word_index.items():
      if word in word2vec.key_to_index and idx < MAX_NB_WORDS:
        embedding_matrix[idx] = word2vec.vectors[word2vec.key_to_index[word]]
```

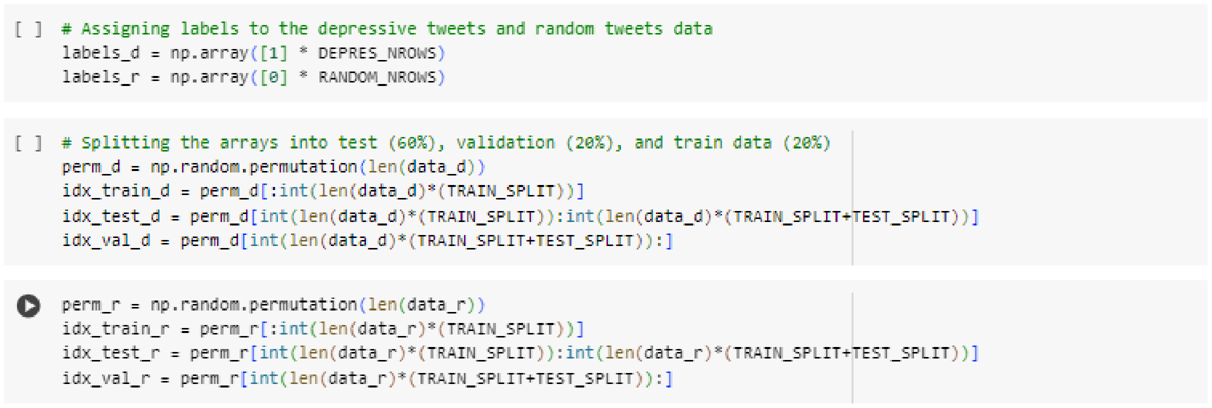
Fig 4.6: Embedding Matrix

If there are fewer unique words than the maximum specified, then n is set to the number of unique words. This ensures that the embedding matrix contains an embedding vector for every word in the vocabulary, while also limiting the size of the matrix.

4.7 Splitting and Formatting:

Assigning labels to the depressive tweets and random tweets data, and splitting the arrays into test (60%), validation (20%), and train data (20%). Combine depressive tweets and random tweets arrays and shuffle.

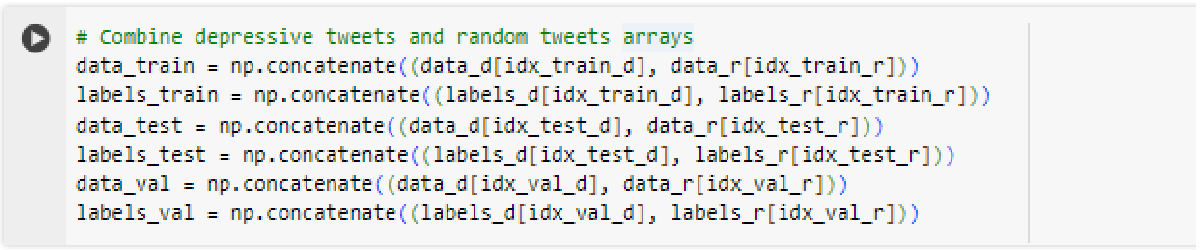
Splitting extract dataset into Training, Testing and Validation for accurate results of it.by combining the both Datasets for increase the accuracy of it based on the format.it will work with the models for perfect accuracy of it.



```
[ ] # Assigning labels to the depressive tweets and random tweets data
    labels_d = np.array([1] * DEPRES_NROWS)
    labels_r = np.array([0] * RANDOM_NROWS)

[ ] # Splitting the arrays into test (60%), validation (20%), and train data (20%)
    perm_d = np.random.permutation(len(data_d))
    idx_train_d = perm_d[:int(len(data_d)*(TRAIN_SPLIT))]
    idx_test_d = perm_d[int(len(data_d)*(TRAIN_SPLIT)):int(len(data_d)*(TRAIN_SPLIT+TEST_SPLIT))]
    idx_val_d = perm_d[int(len(data_d)*(TRAIN_SPLIT+TEST_SPLIT)):]

    perm_r = np.random.permutation(len(data_r))
    idx_train_r = perm_r[:int(len(data_r)*(TRAIN_SPLIT))]
    idx_test_r = perm_r[int(len(data_r)*(TRAIN_SPLIT)):int(len(data_r)*(TRAIN_SPLIT+TEST_SPLIT))]
    idx_val_r = perm_r[int(len(data_r)*(TRAIN_SPLIT+TEST_SPLIT)):]
```

Fig 4.7.1: Splitting the Dataset

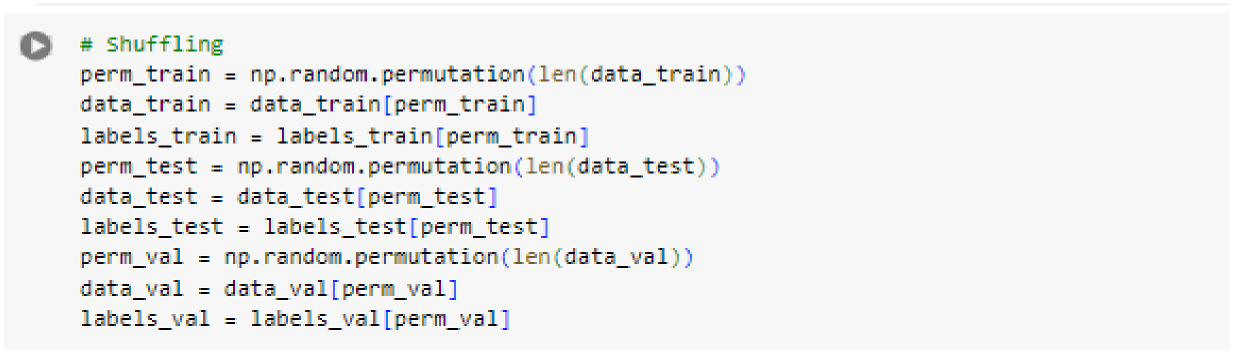
Now we will be combining the both dataset for this code prepares a combined dataset for training, testing, and validation, by combining depressive tweets and random tweets along with their respective labels. It seems to be part of a machine learning or natural language processing task where depressive tweets and random tweets are being used as input data for training a model.



```
# Combine depressive tweets and random tweets arrays
data_train = np.concatenate((data_d[idx_train_d], data_r[idx_train_r]))
labels_train = np.concatenate((labels_d[idx_train_d], labels_r[idx_train_r]))
data_test = np.concatenate((data_d[idx_test_d], data_r[idx_test_r]))
labels_test = np.concatenate((labels_d[idx_test_d], labels_r[idx_test_r]))
data_val = np.concatenate((data_d[idx_val_d], data_r[idx_val_r]))
labels_val = np.concatenate((labels_d[idx_val_d], labels_r[idx_val_r]))
```

Fig 4.7.2: Combining the Datasets

Shuffling the training, testing, and validation datasets along with their corresponding labels. Shuffling is an important step in machine learning to randomize the order of data, preventing the model from learning any spurious patterns based on the order in which data samples are presented. After executing these lines, the training, testing, and validation datasets along with their labels will have their samples shuffled in random order, ensuring that the model does not inadvertently learn from any sequence-related patterns in the data.



```
# Shuffling
perm_train = np.random.permutation(len(data_train))
data_train = data_train[perm_train]
labels_train = labels_train[perm_train]
perm_test = np.random.permutation(len(data_test))
data_test = data_test[perm_test]
labels_test = labels_test[perm_test]
perm_val = np.random.permutation(len(data_val))
data_val = data_val[perm_val]
labels_val = labels_val[perm_val]
```

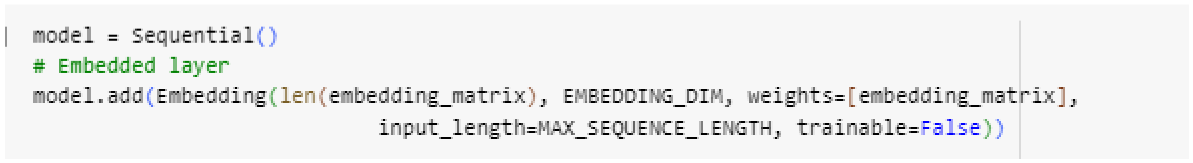
Fig 4.7.3: Shuffling the Data

4.8 Building Model:

The model takes in an input and then outputs a single number representing the probability that the tweet indicates depression. The model takes in each input sentence, replace it with its embedding’s, then run the new embedding vector through a convolutional layer. CNNs are excellent at learning spatial structure from data, the convolutional layer takes advantage of that and learn some structure from the sequential data then pass into a standard LSTM layer. Last but not least, the output of the LSTM layer is fed into a standard dense model for prediction. The layer we using in this model like embedding layer, conventional layer, lstm layer.

4.8.1 Embedding Layer:

An embedding layer is used to convert words or tokens into dense vectors of fixed size. This conversion is crucial because it enables the neural network to represent words in a continuous vector space where similar words are closer to each other.it will use to Dimensionality Reduction, Semantic Representation, Representation Learning, Transfer Learning, Fixed-Length Input. Embedding layers are essential in depression detection tasks (as well as other natural language processing tasks) because they enable the neural network to understand and process textual data in a meaningful way, capturing semantic relationships and contextual information crucial for detecting depression-related language patterns.

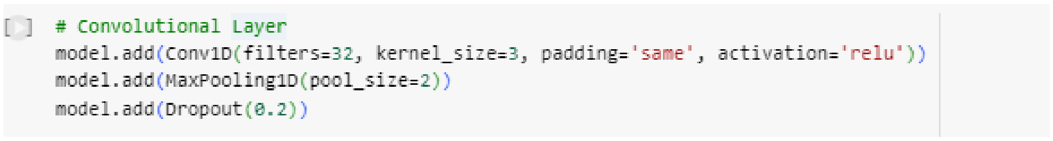


```
model = Sequential()
# Embedded layer
model.add(Embedding(len(embedding_matrix), EMBEDDING_DIM, weights=[embedding_matrix],
                            input_length=MAX_SEQUENCE_LENGTH, trainable=False))
```

Fig 4.8.1: Embedding Layer

4.8.2 Convolutional Layer:

Convolutional layers are typically used in computer vision tasks where spatial relationships between neighbouring pixels are important. However, in recent years, convolutional neural networks (CNNs) have also been applied to natural language processing (NLP) tasks with great success, including depression detection. Here's why convolutional layers are useful in depression detection tasks like Local Feature Extraction, Translation Invariance, Hierarchical Feature Learning, Parameter Sharing, Efficiency, and Regularization. Convolutional layers are useful in depression detection tasks because they excel at capturing local patterns in text data, they are translation invariant, they facilitate hierarchical feature learning, they are computationally efficient, and they can act as a form of regularization. When combined with other layers like LSTM and dense layers, convolutional layers contribute to building effective models for detecting depressive language patterns in text.

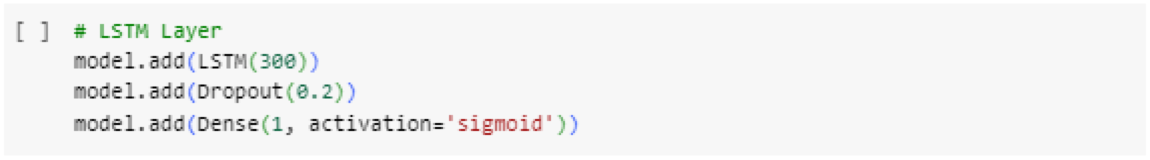


```
[ ]  # Convolutional Layer
     model.add(Conv1D(filters=32, kernel_size=3, padding='same', activation='relu'))
     model.add(MaxPooling1D(pool_size=2))
     model.add(Dropout(0.2))
```

Fig 4.8.2: Convolutional Layer

4.8.3 LSTM Layer:

Long Short-Term Memory (LSTM) layers are a type of recurrent neural network (RNN) architecture specifically designed to capture long-term dependencies in sequential data. In depression detection tasks like Modelling Sequential Dependencies, Handling Variable-Length Sequences, Capturing Contextual Information, Learning Temporal Patterns, Gradient Flow and Backpropagation Through Time, Efficient Representation Learning. LSTM layers are valuable in depression detection tasks because they can effectively model sequential dependencies, handle variable-length input sequences, capture contextual information, learn temporal patterns, address gradient flow issues, and facilitate efficient representation learning from text data. When combined with other layers like convolutional layers and dense layers, LSTM layers contribute to building powerful models capable of detecting depressive language cues in textual data.

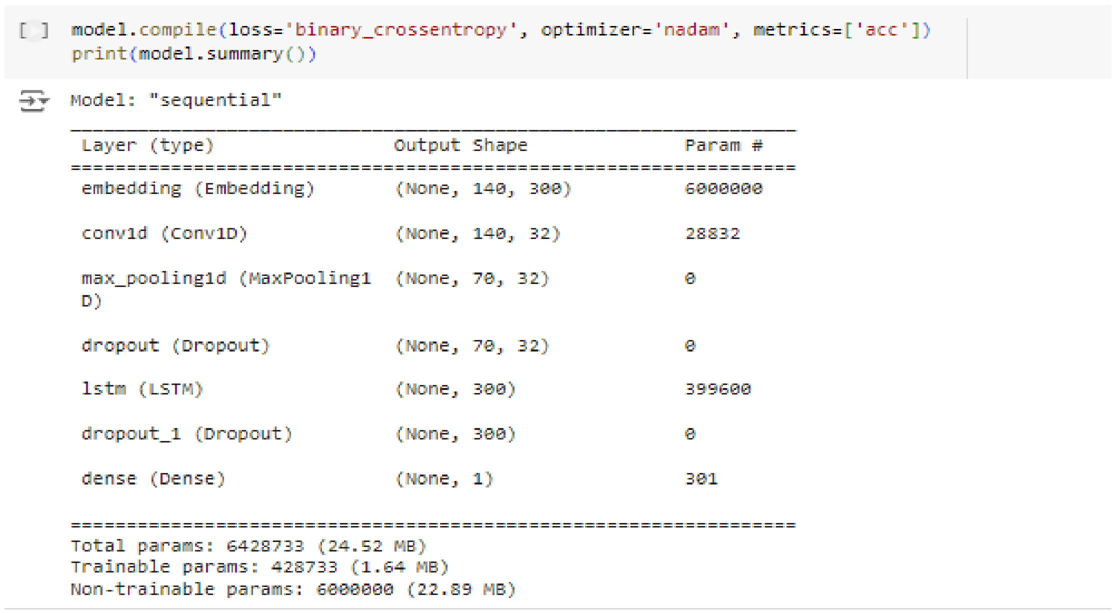


```
[ ]  # LSTM Layer
     model.add(LSTM(300))
     model.add(Dropout(0.2))
     model.add(Dense(1, activation='sigmoid'))
```

Fig 4.8.1: LSTM Layer

4.9 Compile Model:

Compiling the model is an essential step before training it. During compilation, you specify additional parameters necessary for training, such as the loss function, optimizer, and metrics. Here's why compiling the model is necessary, compiling the model allows you to configure these aspects of training, making it ready to be trained on the training data. Once the model is compiled, you can then train it using the fit method by providing the training data, validation data (if any), batch size, and number of epochs. The summary method provides a concise summary of the model architecture, including the types and shapes of the layers, the number of parameters, and the output shape of each layer, which helps in understanding the model's structure and ensuring that it is configured correctly before training. Loss =binary _cross entropy ,optimizer=nadam, metrics=accuracy.



```
model.compile(loss='binary_crossentropy', optimizer='nadam', metrics=['acc'])
print(model.summary())
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 140, 300)          6000000

 conv1d (Conv1D)             (None, 140, 32)           28832

 max_pooling1d (MaxPooling1  (None, 70, 32)            0
 D)

 dropout (Dropout)           (None, 70, 32)            0

 lstm (LSTM)                 (None, 300)               399600

 dropout_1 (Dropout)         (None, 300)               0

 dense (Dense)               (None, 1)                 301

=================================================================
Total params: 6428733 (24.52 MB)
Trainable params: 428733 (1.64 MB)
Non-trainable params: 6000000 (22.89 MB)
```

Fig 4.9: Compile the Model

4.10 Deep Learning and its types:

4.10.1 Deep Learning:

Deep learning is a subset of machine learning, which is a broader field of artificial intelligence (AI). It involves the use of neural networks with multiple layers (deep neural networks) to model and solve complex problems. These neural networks attempt to simulate the behavior of the human brain in order to "learn" from large amounts of data. The term "deep" refers to the multiple layers through which the data is transformed. In traditional machine learning, feature extraction and selection are crucial steps, where humans need to manually identify relevant features for the algorithm to work effectively. In contrast, deep learning algorithms automatically learn hierarchical representations of data through the neural network's layered structure. This enables deep learning models to automatically discover patterns, features, and representations from raw data without explicit programming.

Deep learning has been particularly successful in tasks such as image and speech recognition, natural language processing, and playing strategic games. Notable architectures within deep learning include convolutional neural networks (CNNs) for image-related tasks, recurrent neural network (RNNs) for sequence data, and transformer architectures for natural language processing tasks.

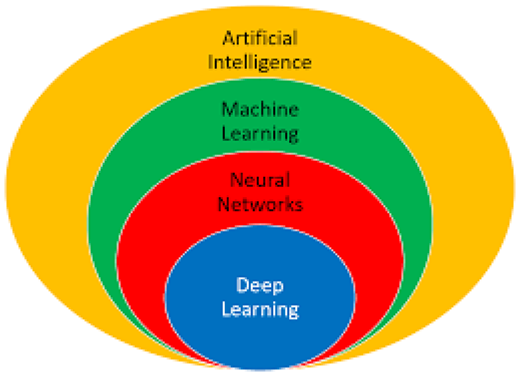


Fig 4.10.1: Subsets of Artificial Intelligence

4.10.2 How does Deep Learning work?

Deep learning works by using artificial neural networks to learn from data. Neural networks are made up of layers of interconnected nodes, and each node is responsible for learning a specific feature of the data. Building on our previous example with images – in an image recognition network, the first layer of nodes might learn to identify edges, the second layer might learn to identify shapes, and the third layer might learn to identify objects.

As the network learns, the weights on the connections between the nodes are adjusted so that the network can better classify the data. This process is called training, and it can be done using a variety of techniques, such as supervised learning, unsupervised learning, and reinforcement learning.

Once a neural network has been trained, it can be used to make predictions with new data it’s received.

4.10.3 NEURAL NETWORKS

A Neural Network is inspired by the structure and function of the human brain. They are a key component of machine learning and artificial intelligence. ANNs consist of interconnected nodes, or artificial neurons, organized into layers. The three main types of layers are the input layer, hidden layers, and output layer.

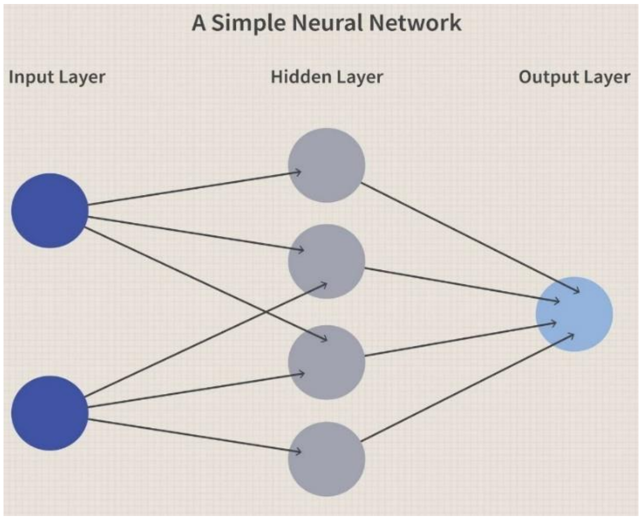


Fig 4.10.3: Structure of Neural Networks

1. **Input Layer**: This layer receives the initial data or input features. Each node in this layer represents a feature or attribute of the input data.

2. **Hidden Layers**: Between the input and output layers, there can be one or more hidden layers. These layers process the input data through a series of weighted connections and activation functions, transforming the input into a form that can be used to make predictions.

3. **Output Layer**: The final layer produces the network's output, which could be in the form of

classifications, predictions, or other relevant information depending on the task (e.g., image recognition, speech recognition, or regression).

Neural networks are versatile and have been successfully applied to a wide range of tasks, including image and speech recognition, natural language processing, and game playing. They are a fundamental component of the broader field of machine learning and artificial intelligence.

4.10.4 Neural Networks - Types:

1. RECURRENT NEURAL NETWORK (RNN)

2. CONVUTIONAL NEURAL NETWORK (CNN)

3. HYBRID DEEP LEARNING MODEL (HDL)

1. RECURRENT NEURAL NETWORK(RNN)

Recurrent Neural Network (RNN) is a type of Neural Network where the output from the previous step is fed as input to the current step. The main and most important feature of RNN is its Hidden state, which remembers some information about a sequence. The state is also referred to as Memory State since it remembers the previous input to the network. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

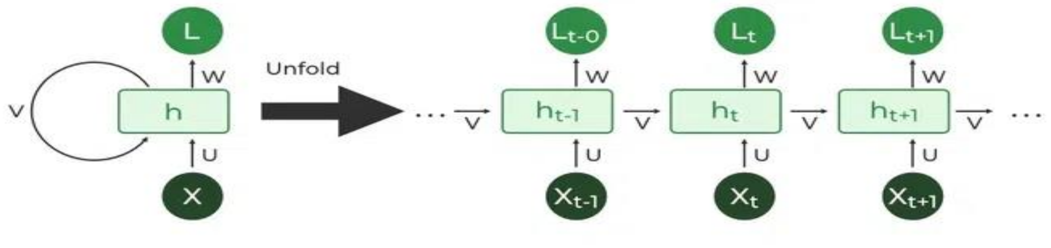


Fig 4.10.4.1: Structure of Recurrent Neural Networks

How RNN Works

The Recurrent Neural Network consists of multiple fixed activation function units, one for each time step. Each unit has an internal state which is called the hidden state of the unit. This hidden state signifies the past knowledge that the network currently holds at a given time step. This hidden state is updated at every time step to signify the change in the knowledge of the network about the past. The hidden state is updated using the following recurrence relation given below are as follows.

The formula for calculating the current state:

$$h_t = f(h_{t-1}, x_t)$$

where:
ht -> current state
ht-1 -> previous state
xt -> input state

Formula for applying Activation function(tanh):

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

where:
whh -> weight at recurrent neuron
wxh -> weight at input neuron

The formula for calculating output:

$$y_t = W_{hy}h_t$$

Where:
Yt -> output
Why -> weight at output layer

2. CONVUTIONAL NEURAL NETWORK

A CNN is a sequence of layers, and every layer of a CNN transforms one volume of activations to another through a differentiable function. What it actually means is that, each layer is associated with converting the information from the values, available in the previous layers, into some more complex information and pass on to the next layers for further generalization.

CNN is a Combination of Two Basic Building Blocks:

1. The Convolution Block: Consists of the Convolution Layer and the Pooling Layer. This layer forms the essential component of Feature-Extraction

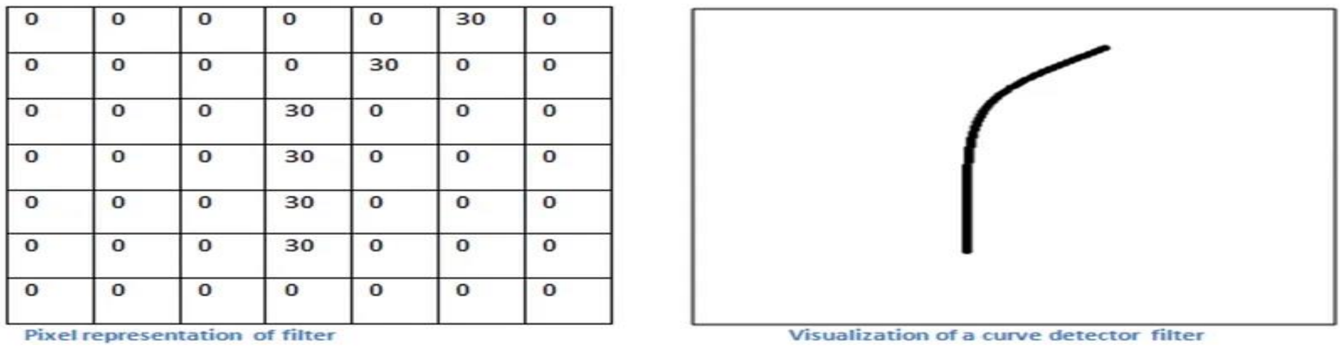
2. The Fully Connected Block: Consists of a fully connected simple neural network architecture. This layer performs the task of Classification based on the input from the convolutional block. We shall define each of these layers now. People who have a little knowledge of image processing might find it easy to grasp, however others can find it easy too.

We shall define each of these layers now. People who have a little knowledge of image processing might find it easy to grasp, however others can find it easy too.

Defining the Layers: We shall describe the working theory of each of the layers.

CONVOLUTIONAL LAYER: The CONVOLUTIONAL LAYER is related to feature extraction. First let us get clear of the idea of 'filters' and 'convolution', then, we shall move on to its implementation in the layer.

Filters: Filters or 'kernels' are also an image that depict a particular feature. For example, let us take the picture of this curve. We take this as a sample feature that we will try to recognize, i.e., determine whether it is present in an image.



0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter
Visualization of a curve detector filter

Fig 4.10.4.2.1: An Example of a Simple Filter Depicting a Curved Line

Convolution: It is a special operation applied on a particular matrix (, usually the image matrix) using another matrix (, usually the filter-matrix). The operation involves multiplying the values of a cell corresponding to a particular row and column, of the image matrix, with the value of the corresponding cell in the filter matrix. We do this for the values of all the cells within the span of the filter matrix and add them together to form an output.

For example:
A (part of the image matrix) = [2 5 17] and B (part of the filter matrix) = [1 0 1]. Then the answer of A *(convolve) B = [2*1 + 5*0 + 17*1] = [2 + 17] = [19]. This is just an example. In practice both of the matrix are 2-D, but the sense of the operation remains the same. This will explain the operation nicely. You can see, how the filter-matrix spans over the whole matrix, performing the operation at each step and generating the desired output matrix:

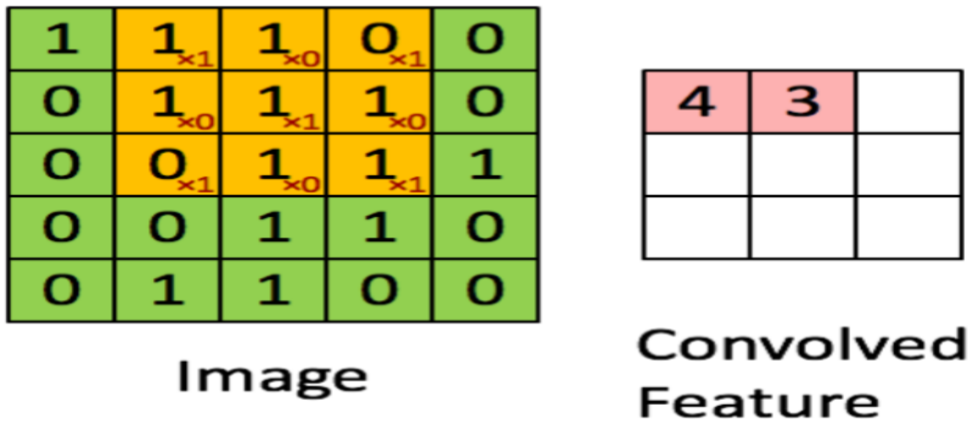
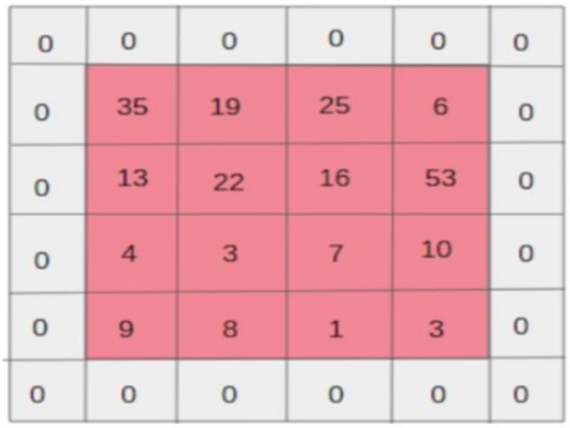


Fig 4.10.4.2.2: Depicts How Convolution Works

Padding:

If we try to visualize the operation of convolution, in our head, as the filter matrix moves over the whole image, we find that the no of times, the values of the cells lying within the matrix is considered for the operation is more than the no. of times, the values of the cells in the corners or at the borders, are accounted for. This implies that the values at the corners or around the borders are not being given equal weightage. To overcome this, we add another row and column, of only 0, at all the sides of the image matrix. This idea is known as padding. In actual sense, these values being '0' wouldn't supply any extra information, but will help into accounting the previously less-accounted for values to be given more weightage.



0	0	0	0	0	0
0	35	19	25	6	0
0	13	22	16	53	0
0	4	3	7	10	0
0	9	8	1	3	0
0	0	0	0	0	0

Fig 4.10.4.2.3: A Padding of 0's around the actual matrix

Striding:

In 'strided' convolution, instead of shifting the filter one-row or onecolumn at a time, we shift it, maybe, 2 or 3 rows or columns, each time. This is generally done to reduce the no of calculation and also reduce the size of the output matrix. For large image, this doesn't results in loss of data, but reduces computation cost on a large scale.

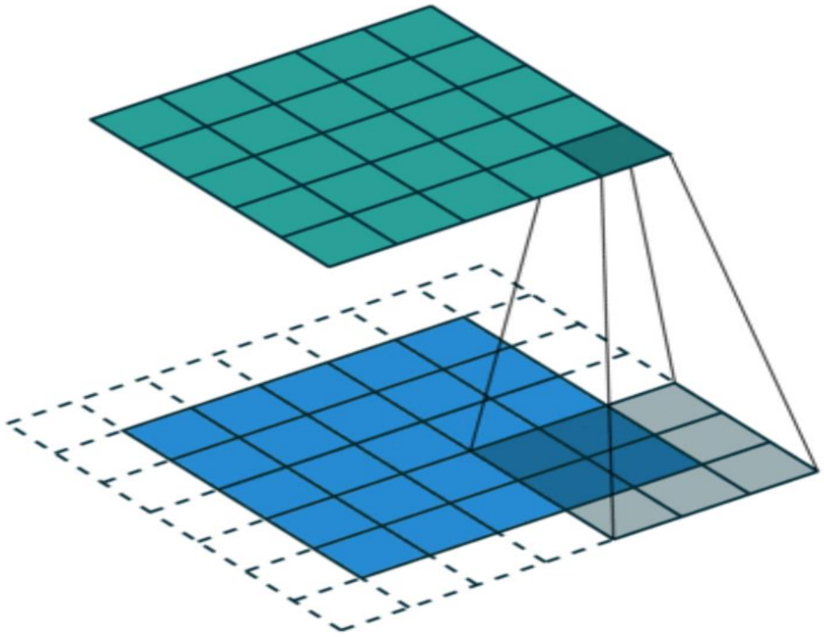


Fig 4.10.4.2.4: This Implementation with stride 2 shows the particular cells that will be Convolved.

RELU Activation:

RELU or Rectified Linear Unit is applied on all the cells of all the output-matrix. The function is defined as:

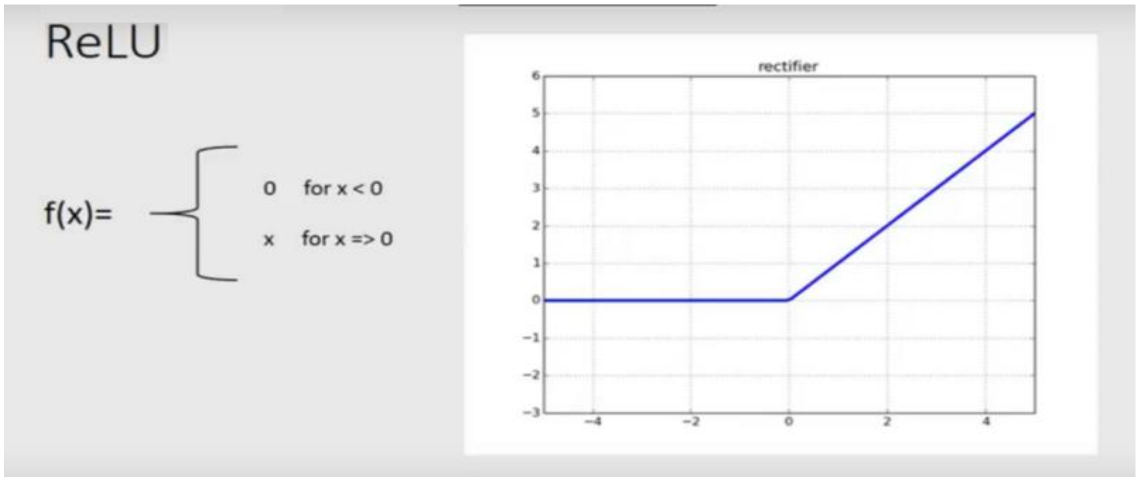


Fig 4.10.4.2.5: Graph of the RELU Function

The basic intuition to derive from here is that, after convolution, if a particular convolution function results in '0' or a negative value, it implies that the feature is not present there and we denote it by '0', and for all the other cases we keep the value. Together with all the operations and the functions applied on the input image, we form the first part of the Convolutional Block.

Pooling Layer:

The Pooling layer consist of performing the process of extracting a particular value from a set of values, usually the max value or the average value of all the values. This reduces the size of the output matrix. For example, for MAX-POOLING, we take in the max value among all the values of say a 2 X 2 part of the matrix. Thus, we are actually taking in the values denoting the presence of a feature in that section of the image. In this way we are getting rid of unwanted information regarding the presence of a feature in a particular portion of the image and considering only what is required to know. It is common to periodically insert a Pooling layer in-between successive convolutional blocks in a CNN architecture. Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network.

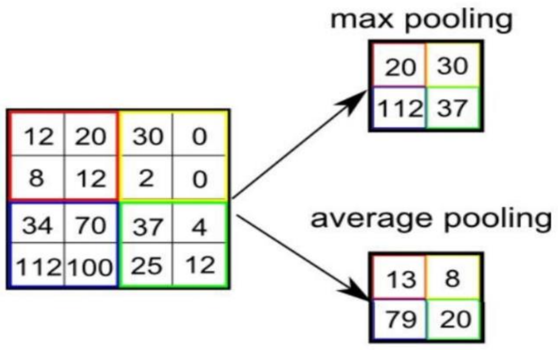


Fig 4.10.4.2.6: An Example of both Max-Pooling and Average-Pooling

Together with the CONVOLUTIONAL LAYER and the POOLING LAYER, we form the CONVOLUTIONAL BLOCK of the CNN architecture. Generally, a simple CNN architecture constitutes of a minimum of three of these Convolutional Block, that performs feature extraction at various levels.

Fully connected layer:

This layer forms the last block of the CNN architecture, related to the task of classification. This is essentially a Fully connected Simple Neural Network, consisting of two or three hidden layers and an output layer generally implemented using 'Softmax Regression', that performs the work of classification among a large no of categories. Hope you can understand the basic idea of the regression analysis from the link.

Bringing Everything Together :

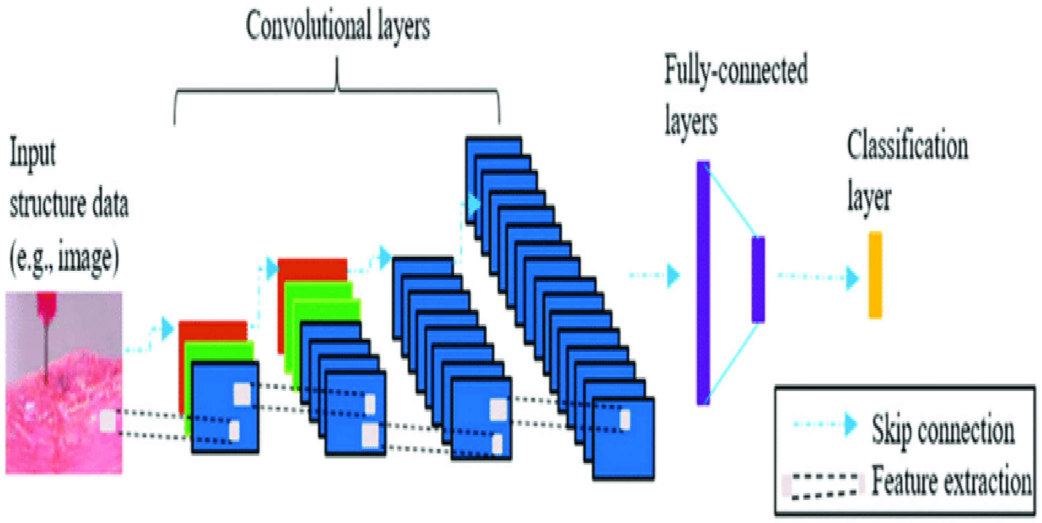


Fig 4.10.4.2.7: Stacking the Concepts under one Roof

3. CONVUTIONAL NEURAL NETWORK-LONG SHORT TERM MEMORY NETWORK(CNN-LSTM)

The CNN Long Short-Term Memory Network or CNN LSTM for short is an LSTM architecture specifically designed for sequence prediction problems with spatial inputs, like images or videos, text input data.

The proposed prediction model is based on a hybrid deep learning (HDL) model using a convolutional neural network (CNN) and a long-term memory (LSTM) as the backbone. An LSTM is a special model that is usually used for time predictions while a CNN network is mainly used for processing images. However, this model is still suitable for prediction. we compare the performance of the CNN-LSTM model are as follows:

- We present and analyse a hybrid deep learning model for predicting the TEXT.
- We compare and evaluate the performance of the proposed HDL model (CNN-LSTM versus LSTM-CNN) with baseline models (CNN only and LSTM only) for prediction.
- We investigate how textual data can influence the prediction result to determine the sequential data then pass into a standard LSTM layer.
- We investigate how the HDL model behaves when the input data used has a different size and different parameters.
- The output of the LSTM layer is fed into a standard dense model for prediction.

3.1ARCHTECTURE:

This architecture has also been used on speech recognition and natural language processing problems where CNNs are used as feature extractors for the LSTMs on audio and textual input data.

This architecture is appropriate for problems that:

- Have spatial structure in their input such as the 2D structure or pixels in an image or the 1D structure of words in a sentence, paragraph, or document.
- Have a temporal structure in their input such as the order of images in a video or words in text, or require the generation of output with temporal structure such as words in a textual description.

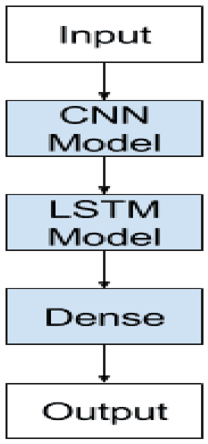


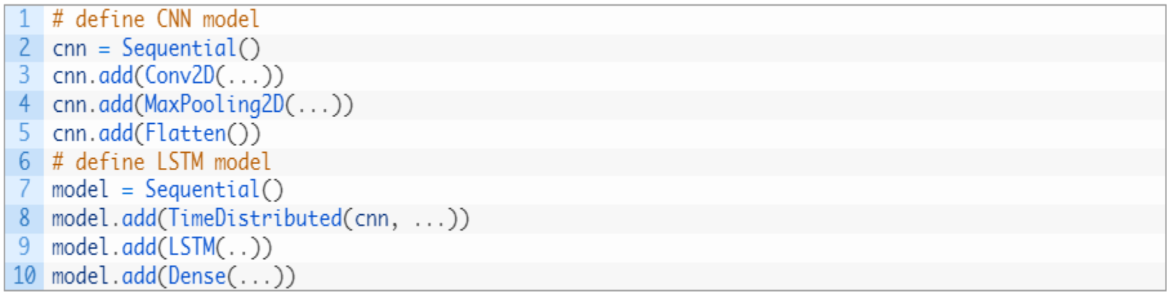
Fig 4.10.4.2.8: Flow of Data Through the Model

3.2 CNN-LSTM MODEL

We can define a CNN LSTM model in Keras by first defining the CNN layer or layers, wrapping them in a Time Distributed layer and then defining the LSTM and output layers.

We have two ways to define the model that are equivalent and only differ as a matter of taste.

You can define the CNN model first, then add it to the LSTM model by wrapping the entire sequence of CNN layers in a Time Distributed layer, as follows:



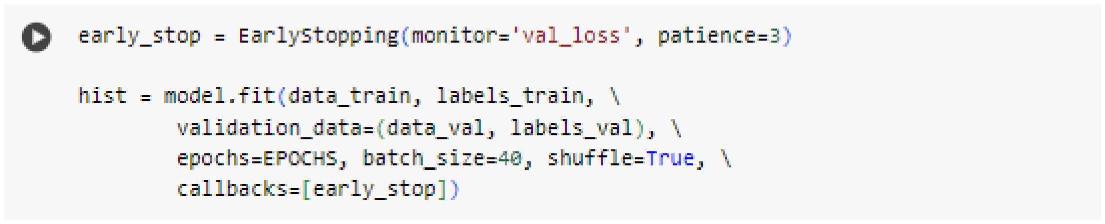
```
# define CNN model
cnn = Sequential()
cnn.add(Conv2D(...))
cnn.add(MaxPooling2D(...))
cnn.add(Flatten())
# define LSTM model
model = Sequential()
model.add(TimeDistributed(cnn, ...))
model.add(LSTM(..))
model.add(Dense(...))
```

Fig 4.10.4.2.9: Example of CNN-LSTM Model

Therefore A CNN-LSTM architecture has wide-ranging applications as it stands at the helm of deep learning and Natural Language Processing. It allows us to use state of the art neural models for NLP tasks such as the transformer for sequential image and video data. At the same time, extremely powerful CNN networks can be used for sequential data such as the natural language. Hence, it allows us to leverage the useful aspects of powerful models in tasks they have never been used for before.

5. TRAINING THE MODEL:

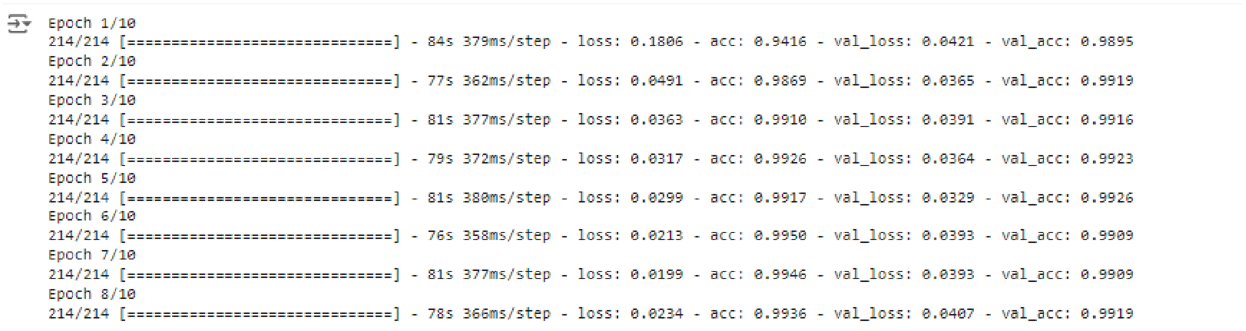
The model is trained EPOCHS time, and Early Stopping argument is used to end training if the loss or accuracy don't improve within 10 epochs. Once the training parameters are defined and the trainer is selected, the next phase is to actually train the model. In training we take the tokenized values from feature extraction phase and the training arguments to start training. The training is done in epochs. Each epoch consists of 10batches.



```
early_stop = EarlyStopping(monitor='val_loss', patience=3)

hist = model.fit(data_train, labels_train, \
        validation_data=(data_val, labels_val), \
        epochs=EPOCHS, batch_size=40, shuffle=True, \
        callbacks=[early_stop])
```

Fig 5.0.1: Early Stopping

Once those batches are done a epoch is said to be completed. As we have selected logging strategy and evaluation strategy as epoch after every epoch the model is evaluated and loss of the model is calculated.



```
Epoch 1/10
214/214 [==============================] - 84s 379ms/step - loss: 0.1806 - acc: 0.9416 - val_loss: 0.0421 - val_acc: 0.9895
Epoch 2/10
214/214 [==============================] - 77s 362ms/step - loss: 0.0491 - acc: 0.9869 - val_loss: 0.0365 - val_acc: 0.9919
Epoch 3/10
214/214 [==============================] - 81s 377ms/step - loss: 0.0363 - acc: 0.9910 - val_loss: 0.0391 - val_acc: 0.9916
Epoch 4/10
214/214 [==============================] - 79s 372ms/step - loss: 0.0317 - acc: 0.9926 - val_loss: 0.0364 - val_acc: 0.9923
Epoch 5/10
214/214 [==============================] - 81s 380ms/step - loss: 0.0299 - acc: 0.9917 - val_loss: 0.0329 - val_acc: 0.9926
Epoch 6/10
214/214 [==============================] - 76s 358ms/step - loss: 0.0213 - acc: 0.9950 - val_loss: 0.0393 - val_acc: 0.9909
Epoch 7/10
214/214 [==============================] - 81s 377ms/step - loss: 0.0199 - acc: 0.9946 - val_loss: 0.0393 - val_acc: 0.9909
Epoch 8/10
214/214 [==============================] - 78s 366ms/step - loss: 0.0234 - acc: 0.9936 - val_loss: 0.0407 - val_acc: 0.9919
```

Fig 5.0.2: Training the Model

Once all the epochs are done and the loss is calculated then we move on to evaluation of the model, where we use different evaluation strategies to evaluate the working of the model.

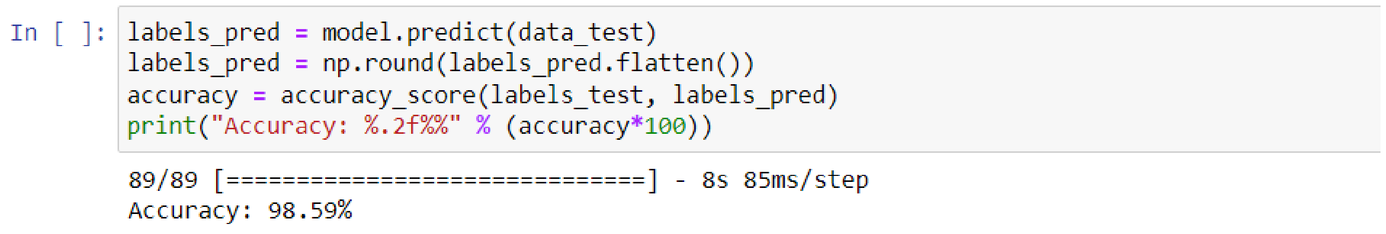
5.1 EVALUATION METRICS:

Evaluation metrics can help you assess your model's performance, monitor your ML system in Production, and control your model to fit your business needs. Our goal is to create and select a model which gives high accuracy on out-of-sample data. It's Very crucial to use multiple evaluation metrics to evaluate your model because a model may Perform well using one measurement from one evaluation metric while may perform poorly.
Using another measurement from another evaluation metric.

5.1.1 ACCURACY:

Accuracy of an algorithm is represented as the ratio of correctly classified predictions (TP+TN) to the total number of predictions (TP+TN+FP+FN).

Accuracy = (TP+TN) / (TP+TN+FP+FN)



```
In [ ]: labels_pred = model.predict(data_test)
        labels_pred = np.round(labels_pred.flatten())
        accuracy = accuracy_score(labels_test, labels_pred)
        print("Accuracy: %.2f%%" % (accuracy*100))

        89/89 [==============================] - 8s 85ms/step
        Accuracy: 98.59%
```

Fig 5.1.1: Model Accuracy

5.1.2 PRECISION:

Precision is the ability of a classifier not to label an instance positive that is actually negative. For each class it is defined as the ratio of true positives to the sum of true and false positives.

• TP – True Positives

• FP – False Positives

Precision: Accuracy of positive predictions.

Precision= TP / (TP+FP)

5.1.3 RECALL:

Recall is the ability of a classifier to find all positive instances. For each class it is defined as the ratio of true positives to the sum of true positives and false negatives.
Recall: Fraction of positives that were correctly identified.

Recall = TP / (TP+FN)

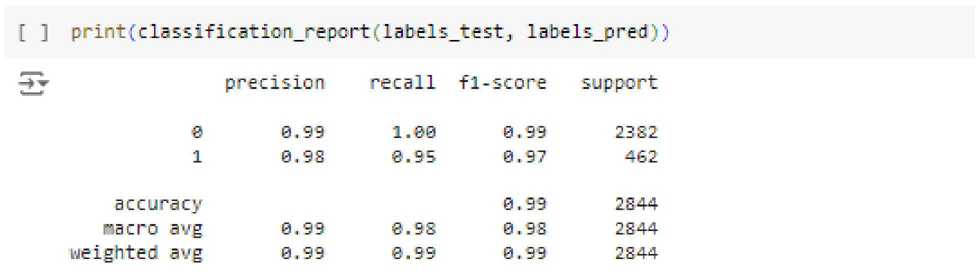
5.1.4 F1 SCORE:

The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0. Generally speaking, F1 scores are lower than accuracy measures as they embed precision and recall into their computation. As a rule of thumb, the weighted average of F1 should be used to compare classifier models, not global accuracy.

$$\text{F1 Score} = 2*(\text{Recall}*\text{precision}) / (\text{Recall}+\text{Precision})$$

6. RESULT AND ANALYSIS

6.1 Evaluation Metrics:



```
[ ] print(classification_report(labels_test, labels_pred))

              precision    recall  f1-score   support

           0       0.99      1.00      0.99      2382
           1       0.98      0.95      0.97       462

    accuracy                           0.99      2844
   macro avg       0.99      0.98      0.98      2844
weighted avg       0.99      0.99      0.99      2844
```

Fig 6.1: Evaluation Metrics

6.1.1 TRAINING AND VALIDATION ACCURACY

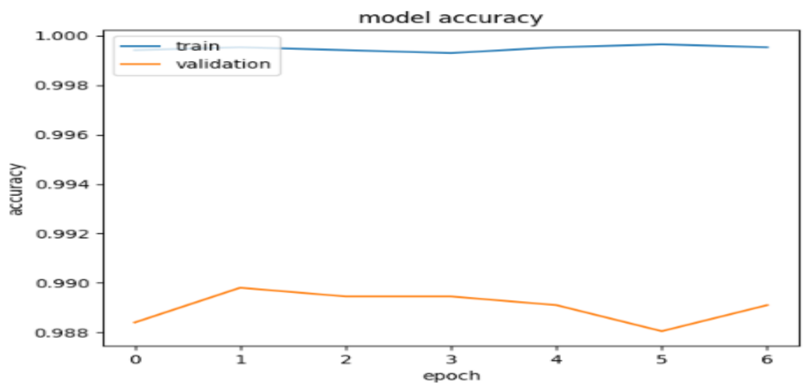


Fig 6.1.1: Training and Validation Accuracy

6.1.2 TRAINING LOSS AND TESTING LOSS

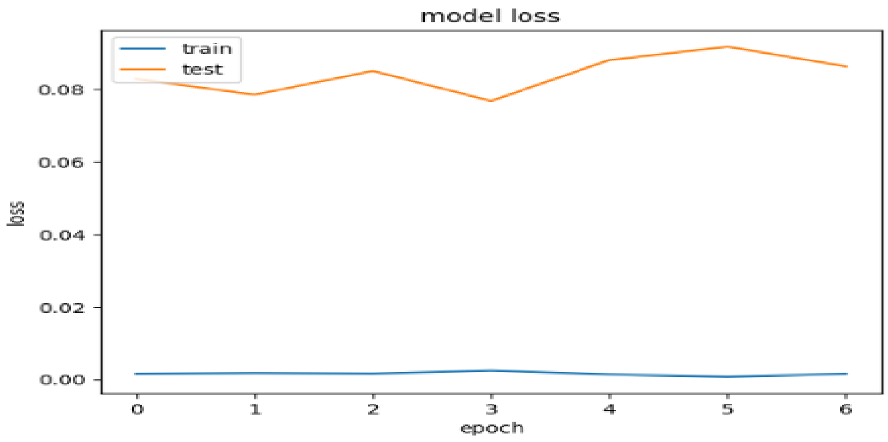
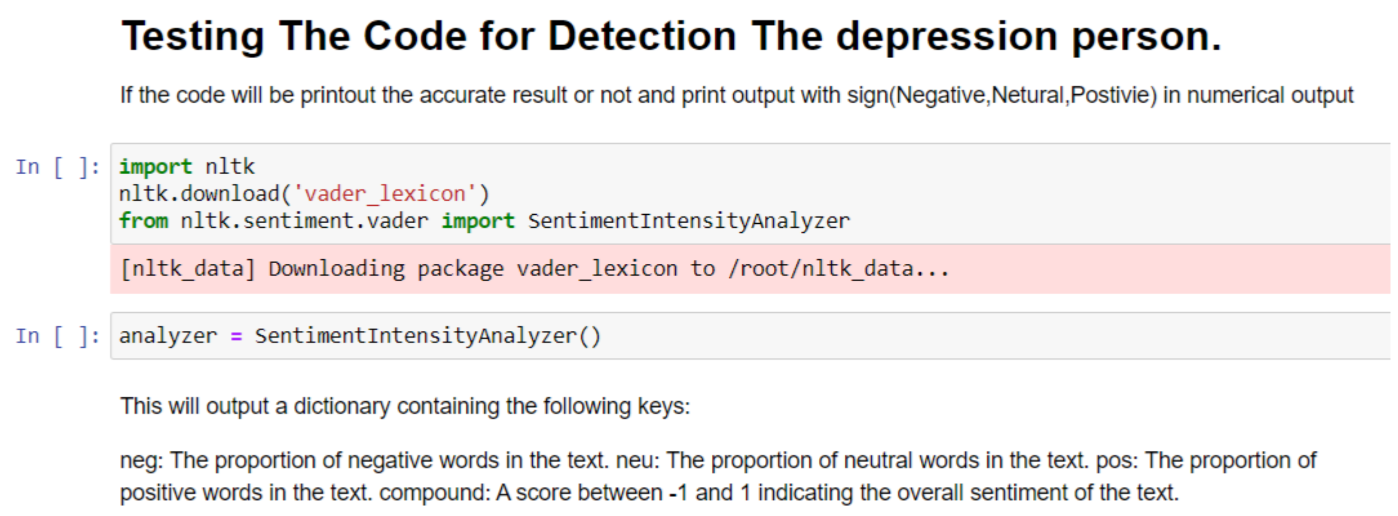


Fig 6.1.2: Training and Testing Loss

7. SCREENSHOTS



Testing The Code for Detection The depression person.

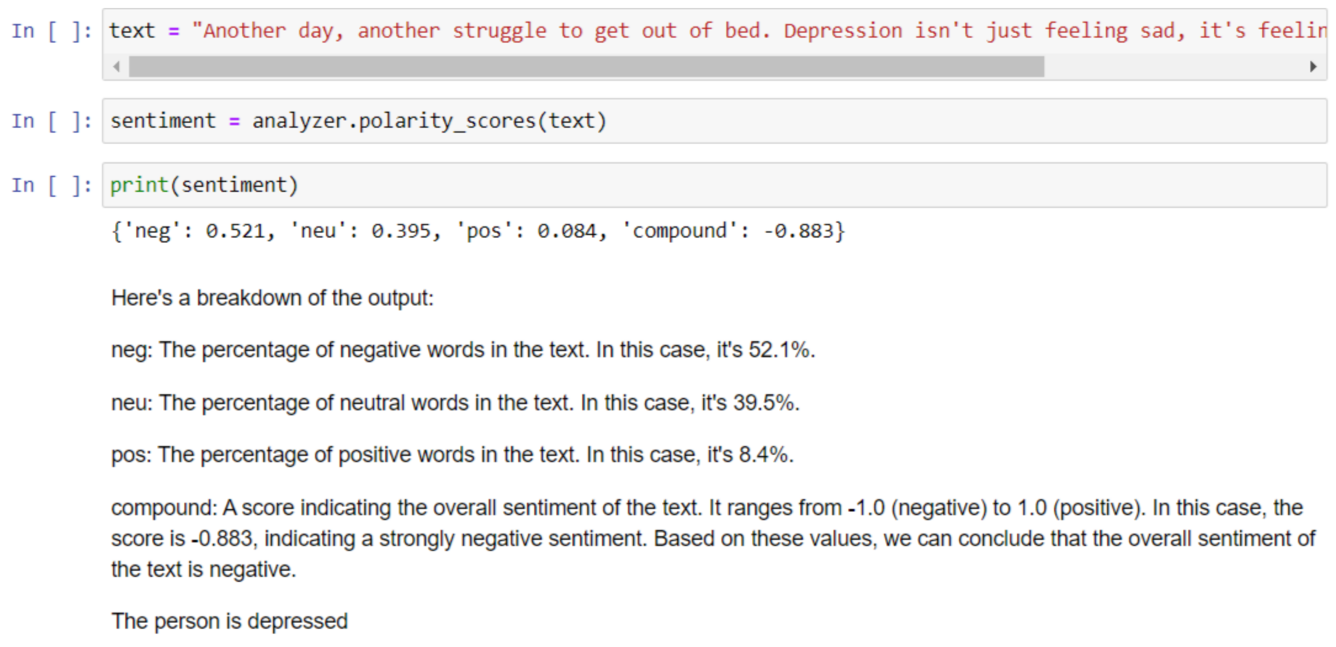
If the code will be printout the accurate result or not and print output with sign(Negative,Netural,Postivie) in numerical output

```
In [ ]: import nltk
        nltk.download('vader_lexicon')
        from nltk.sentiment.vader import SentimentIntensityAnalyzer

        [nltk_data] Downloading package vader_lexicon to /root/nltk_data...

In [ ]: analyzer = SentimentIntensityAnalyzer()
```

This will output a dictionary containing the following keys:

neg: The proportion of negative words in the text. neu: The proportion of neutral words in the text. pos: The proportion of positive words in the text. compound: A score between -1 and 1 indicating the overall sentiment of the text.

Fig 7.1: Importing Sentiment Intensity Analyzer



```
In [ ]: text = "Another day, another struggle to get out of bed. Depression isn't just feeling sad, it's feelin

In [ ]: sentiment = analyzer.polarity_scores(text)

In [ ]: print(sentiment)

        {'neg': 0.521, 'neu': 0.395, 'pos': 0.084, 'compound': -0.883}
```

Here's a breakdown of the output:

neg: The percentage of negative words in the text. In this case, it's 52.1%.

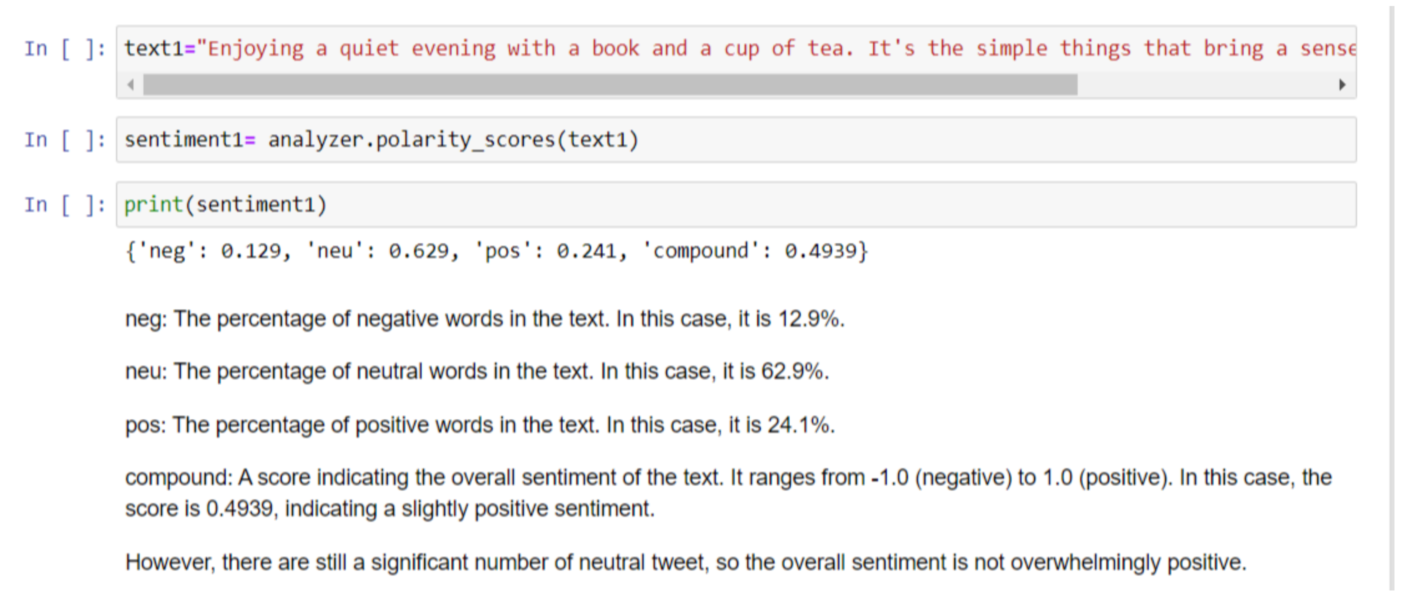
neu: The percentage of neutral words in the text. In this case, it's 39.5%.

pos: The percentage of positive words in the text. In this case, it's 8.4%.

compound: A score indicating the overall sentiment of the text. It ranges from -1.0 (negative) to 1.0 (positive). In this case, the score is -0.883, indicating a strongly negative sentiment. Based on these values, we can conclude that the overall sentiment of the text is negative.

The person is depressed

Fig 7.2: Printing 'Sentiment'



```
In [ ]: text1="Enjoying a quiet evening with a book and a cup of tea. It's the simple things that bring a sense
```

```
In [ ]: sentiment1= analyzer.polarity_scores(text1)
```

```
In [ ]: print(sentiment1)
        {'neg': 0.129, 'neu': 0.629, 'pos': 0.241, 'compound': 0.4939}
```

neg: The percentage of negative words in the text. In this case, it is 12.9%.

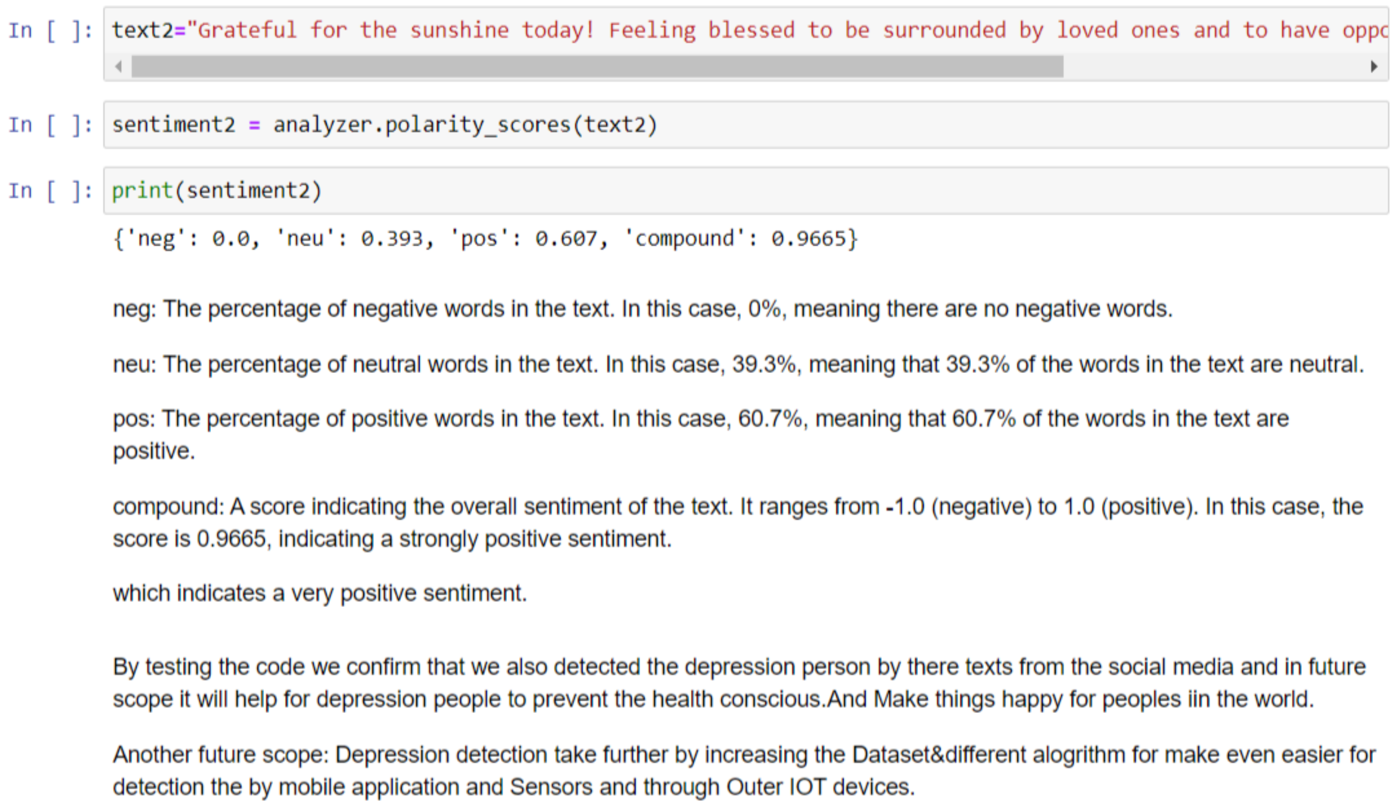
neu: The percentage of neutral words in the text. In this case, it is 62.9%.

pos: The percentage of positive words in the text. In this case, it is 24.1%.

compound: A score indicating the overall sentiment of the text. It ranges from -1.0 (negative) to 1.0 (positive). In this case, the score is 0.4939, indicating a slightly positive sentiment.

However, there are still a significant number of neutral tweet, so the overall sentiment is not overwhelmingly positive.

Fig 7.1: Printing 'Sentiment1'



```
In [ ]: text2="Grateful for the sunshine today! Feeling blessed to be surrounded by loved ones and to have oppo
```

```
In [ ]: sentiment2 = analyzer.polarity_scores(text2)
```

```
In [ ]: print(sentiment2)
        {'neg': 0.0, 'neu': 0.393, 'pos': 0.607, 'compound': 0.9665}
```

neg: The percentage of negative words in the text. In this case, 0%, meaning there are no negative words.

neu: The percentage of neutral words in the text. In this case, 39.3%, meaning that 39.3% of the words in the text are neutral.

pos: The percentage of positive words in the text. In this case, 60.7%, meaning that 60.7% of the words in the text are positive.

compound: A score indicating the overall sentiment of the text. It ranges from -1.0 (negative) to 1.0 (positive). In this case, the score is 0.9665, indicating a strongly positive sentiment.

which indicates a very positive sentiment.

By testing the code we confirm that we also detected the depression person by there texts from the social media and in future scope it will help for depression people to prevent the health conscious.And Make things happy for peoples iin the world.

Another future scope: Depression detection take further by increasing the Dataset&different alogrithm for make even easier for detection the by mobile application and Sensors and through Outer IOT devices.

Fig 7.3: Printing 'Sentiment2'

8. CONCLUSION:

Globally, millions of people suffer from a complicated mental illness known as depression, which stems from complex interactions among biological, psychological, and environmental factors. The ability to offer timely support and treatment can be greatly enhanced by recognizing depression early and knowing its causes, thereby increasing the quality of life of those affected.

The project developed a model for detecting depression in social media texts using the LSTM method. The model achieved an accuracy of about 0.98%, which was a considerable performance improvement. The system's effectiveness in accurately identifying depression signs was further tested through precision, recall, and F1-score, among other measures; it scored well on all these metrics.

FUTURE SCOPE:

Improving the performance of LSTM model with complex algorithms like BERT or Ensemble methods, for dealing with various forms of text. With these changes, this model could become extremely helpful in identifying mental disorders at an early stage, thereby providing the necessary help on time and perhaps improving their overall mental health.

9. REFERENCES:

1. Islam, Md Rafiqul, et al. "Depression detection from social network data using machine learning techniques." Health information science and systems 6 (2018): 1-12..

2. Cui, Jingfeng, et al. "Survey on sentiment analysis: evolution of research methods and topics." Artificial Intelligence Review 56.8 (2023): 8469-8510.

3. Fan, Weiguo, and Michael D. Gordon. "The power of social media analytics." Communications of the ACM 57.6 (2014): 74-81.

4. Chiong, Raymond, et al. "A textual-based featuring approach for depression detection using machine learning classifiers and social media texts." Computers in Biology and Medicine 135 (2021): 104499.

5. Ravishankara, K., Vaisakh Dhanush, and I. S. Srajan. "Whatsapp chat analyzer." International Journal of Engineering Research & Technology 9.5 (2020): 897-900.

6. Sai, T. Siva Ratna, et al. "CHAT ANALYSIS ON WHATSAPP USING MACHINE LEARNING." Journal of Engineering Sciences 14.04 (2023).

7. Jadhav, Aditya, et al. "WhatsApp chat sentiment analyzer." International Journal for Research in Applied Science and Engineering Technology 10.12 (2022): 512-516.

8. Iduh, B. Nwamaka. "WhatsApp Network Group Chat Analysis Using Python Programming." International Journal of Latest Technology in Engineering, Management & Applied Science (IJLTEMAS) 9.02 (2020): 15..

9. Garg, Deepak, et al., eds. Advanced Computing: 10th International Conference, IACC 2020, Panaji, Goa, India, December 5–6, 2020, Revised Selected Papers, Part I. Vol. 1367. Springer Nature, 2021.

10. Ahmad, Abubakar, Abdulhafiz A. Nuhu, and Abdulkadir Abubakar. "A Comprehensive Data Analysis on FUDMA ASUU Whatsapp Group Chat." FUDMA Journal of Sciences 5.2 (2021): 26-33..

11. Nandwani, Pansy, and Rupali Verma. "A review on sentiment analysis and emotion detection from text." Social network analysis and mining 11.1 (2021): 81.

12. Drus, Zulfadzli, and Haliyana Khalid. "Sentiment analysis in social media and its application: Systematic literature review." Procedia Computer Science 161 (2019): 707-714.

13. Bhardwaj, Aditya. "Sentiment Analysis and Text Classification for Social Media Contents Using Machine Learning Techniques." Proceedings of the 2nd International Conference on IoT, Social,

Mobile, Analytics & Cloud in Computational Vision & Bio-Engineering (ISMAC-CVB 2020). 2020.

14. Jan, Ali. "NLP and Sentiment Analysis: The Good, The Bad, and In-between."

15. Dalmia, Ayushi, Manish Gupta, and Vasudeva Varma. "IIIT-H at SemEval 2015: Twitter sentiment analysis–the good, the bad and the neutral!." Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015). 2015.
16. Alsayat, Ahmed. "Improving sentiment analysis for social media applications using an ensemble deep learning language model." Arabian Journal for Science and Engineering 47.2 (2022): 2499-2511.

17. Babu, Nirmal Varghese, and E. Grace Mary Kanaga. "Sentiment analysis in social media data for depression detection using artificial intelligence: a review." SN computer science 3.1 (2022): 74.

18. Priya, Anu, Shruti Garg, and Neha Prerna Tigga. "Predicting anxiety, depression and stress in modern life using machine learning algorithms." Procedia Computer Science 167 (2020): 1258-1267.

19. Khan, Md Rafidul Hasan, et al. "Sentiment analysis from Bengali depression dataset using machine learning." 2020 11th international conference on computing, communication and networking technologies (ICCCNT). IEEE, 2020.

20. Fan, Weiguo, and Michael D. Gordon. "The power of social media analytics." Communications of the ACM 57.6 (2014): 74-81.
21. Bokolo, Biodoumoye George, and Qingzhong Liu. "Deep learning-based depression detection from social media: Comparative evaluation of ml and transformer techniques." Electronics 12.21 (2023): 4396.
22. Safa, Ramin, Peyman Bayat, and Leila Moghtader. "Automatic detection of depression symptoms in twitter using multimodal analysis." The Journal of Supercomputing 78.4 (2022): 4709-4744..

23. Havigerová, Jana M., et al. "Text-based detection of the risk of depression." Frontiers in psychology 10 (2019): 385113.

24. Hossain, Md Tazmim, Md Arafat Rahman Talukder, and Nusrat Jahan. "Depression prognosis using natural language processing and machine learning from social media status." International Journal of Electrical and Computer Engineering 12.3 (2022): 2847.

25. Alm, Cecilia Ovesdotter, Dan Roth, and Richard Sproat. "Emotions from text: machine learning for text-based emotion prediction." Proceedings of human language technology conference and conference on empirical methods in natural language processing. 2005..

26. Angskun, Jitimon, Suda Tipprasert, and Thara Angskun. "Big data analytics on social networks for real-time depression detection." Journal of Big Data 9.1 (2022): 69.

27. Gratch, Jonathan, et al. "The distress analysis interview corpus of human and computer interviews." LREC. 2014.

28. Girard, Jeffrey M., and Jeffrey F. Cohn. "Automated audiovisual depression analysis." Current opinion in psychology 4 (2015): 75-79.

29. Ma, Xingchen, et al. "Depaudionet: An efficient deep model for audio based depression classification." Proceedings of the 6th international workshop on audio/visual emotion challenge. 2016.

APPENDIX

A. Sample Code:

```
!pip install twint
!pip install ftfy
import warnings
warnings.filterwarnings("ignore")
import ftfy
import matplotlib.pyplot as plt
import nltk
import numpy as np
import pandas as pd
import re
from math import exp
from numpy import sign
from sklearn.metrics import  classification_report, confusion_matrix, accuracy_score
from gensim.models import KeyedVectors
from nltk.corpus import stopwords
from nltk import PorterStemmer
from keras.models import Model, Sequential
from keras.callbacks import EarlyStopping, ModelCheckpoint
from keras.layers import Conv1D, Dense, Input, LSTM, Embedding, Dropout, Activation, MaxPooling1D
from keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
# Reproducibility
np.random.seed(1234)

DEPRES_NROWS = 3200  # number of rows to read from DEPRESSIVE_TWEETS_CSV
RANDOM_NROWS = 12000 # number of rows to read from RANDOM_TWEETS_CSV
MAX_SEQUENCE_LENGTH = 140 # Max tweet size
MAX_NB_WORDS = 20000
EMBEDDING_DIM = 300
TRAIN_SPLIT = 0.6
TEST_SPLIT = 0.2
LEARNING_RATE = 0.1
EPOCHS= 10
DEPRESSIVE_TWEETS_CSV = '/content/drive/MyDrive/Data Sets/depressive_tweets_processed.csv'
RANDOM_TWEETS_CSV = '/content/drive/MyDrive/Data Sets/Dataset.csv'
EMBEDDING_FILE = '/content/drive/MyDrive/Data Sets/GoogleNews-vectors-negative300.bin.gz'
depressive_tweets_df = pd.read_csv(DEPRESSIVE_TWEETS_CSV, sep = '|', header = None, usecols = range(0,9), nrows = DEPRES_NROWS)
random_tweets_df = pd.read_csv(RANDOM_TWEETS_CSV, encoding = "ISO-8859-1", usecols = range(0,4), nrows = RANDOM_NROWS)
```

B. SOFTWARE AND HARDWARE REQUIREMENTS

SOFTWARE REQUIREMENTS:

Operating System	:	Window 10 or above , Linux
Technology	:	Python
IDE	:	Jupyter Notebook
Python Version	:	Python 3.9 or above

HARDWARE REQUIREMENTS:

CPU	:	Intel i5 or Ryzen 5 or higher
GPU	:	GTX 1650 or higher
Speed	:	2.6 GHz
RAM	:	8 GB

C. TECHNOLOGY USED

Python:

Python was created by Guido van Rossum in the late 1980s as a high-level interpreted programming language that gained popularity due to its simplicity and readability. With vast libraries and frameworks, it is used globally for activities ranging from web development to data science.

Python's flexibility and adaptability to different systems is another selling point; this means that it can be used effectively in various kinds of environments, such as Windows, MacOS X, and UNIX-based systems, among others, thus reaching out to many people. Further, Python also merges easily with other languages, thereby allowing programmers to create modules in C/C++ or Java, which they can then integrate into their code base.

The standard library of Python provides numerous modules and functions across file manipulation, network programming, and web development. With this set of tools and packages offered through the Python Package Index (PyPI), which is a repository for 3rd-party software extensions, you can build web applications quickly. Python is well-equipped for data science and machine learning. NumPy, Pandas, or Matplotlib are some libraries that enable easy data manipulation, analysis, and visualization; Tensor Flow and PyTorch, among others, serve as powerful frameworks for building training models used in artificial intelligence.

Pandas:

Pandas belong to an amazing manipulation library that was made by Python open source. If you're dealing with tabular data sets or time series data that are structured in nature, this is the most efficient because it provides easy-to-use data structures as well as some useful tools for data analysis. This program was created by Wes McKinney in 2008 and has been widely adopted among data scientists due to its flexibility and comprehensive functionality. The two primary components of Pandas include the series and the data Frame. A series is a labeled array and can hold data of any type in one dimension. It's similar to a column in an Excel spreadsheet or a single column from an SQL table. On the other hand, a data frame is a labeled data structure with two dimensions; it resembles a table or spreadsheet where each column may contain different kinds of information. These are useful for working with organized data and performing complex transformations on it.
Pandas have several attributes that help you work with data. You can use it to deal with missing values, filter data based on some criteria, arrange information in ascending or descending order according to one or more columns, align

different datasets together on a shared set of labels, and perform various mathematical operations like adding or multiplying all elements in a row by the same number.

All these tasks and many others can be accomplished in Pandas through an easy-to-understand syntax, thereby simplifying the manipulation of datasets. One of the areas where Pandas excels is in its handling of time series data. It includes various functionalities for time index creation and manipulation; any form of resampling over different periods; and conversions between time zones, among others, thus making it very useful when dealing with datasets containing dated observations such as stock prices, climate measurements, or readings from sensors distributed across different locations.

NumPy:

NumPy is a powerful open-source library for numerical computation in Python. It provides array objects that are multidimensional and high-performance, as well as tools for working with these arrays efficiently and many mathematical functions. Scientific computing and data analysis with Python cannot be done without NumPy.

The primary object in NumPy is the ‘array’, which can be interpreted as an ‘n-dimensional array’. This means that it is a container that can only hold elements of the same data type. Thus, NumPy enables us to store large datasets in memory-efficient ways. In addition, ndarrays allow for array-oriented arithmetic operations and manipulation of large datasets, which would be too slow to do in Python. Instead, logical operations are done on an entire array, so they are much faster than using loops over elements one by one. NumPy relies heavily on vectorized operations; this greatly speeds up computations, as many operations can be performed without even changing their storage location. You can use these operations to work with arrays that only have one row, as well as matrices with multiple dimensions, where every item might have one or multiple index numbers depending on the number of dimensions present in it.
NumPy has many tools for creating, reshaping, and manipulating arrays, including functions and methods. It can work with several data types, such as integers, floats, and complex numbers. NumPy arrays may be altered in shape, sliced, or indexed, which enables efficient data extraction and manipulation.

The ability to do vectorized operations on arrays is one of the main advantages of using NumPy. Code becomes faster and shorter because vectorization eliminates the need for explicit loops. NumPy has different mathematical functions that can be used element-wise on arrays, like arithmetic operations, trigonometric functions, exponential or logarithmic functions, and so on, which speed up numerical computations remarkably.
One of the tools provided by the NumPy library to work with arrays is array broadcasting. This is a software pattern that allows for efficient operations on arrays with different shapes or sizes. It does this by aligning their dimensions,

which eliminates the need for explicit array expansion or replication, thereby simplifying the code’s readability and efficiency.

NLTK:

Natural Language Toolkit, best known as NLTK, is a Python library for human language processing that is open source. In 2001, Steven Bird and Edward Loper developed it, which has been widely used by NLP researchers, educators, and industry professionals. NLTK contains tools, data, and processing algorithms to assist machines in understanding human languages, whether they are in written or spoken form.

NLTK has a variety of features for different NLP tasks. Tokenization and stemming are just a few of the many available functionalities in NLTK for natural language processing. With these tools, developers can convert raw texts into something else that can be analyzed or even modeled.

The Natural Language Toolkit (NLTK) is known for having many corpora. These are large bodies of text used to train models and algorithms in natural language processing. There are several such sets available with NLTK, including (but not limited to) the Gutenberg Corpus, Brown Corpus, Reuters Corpus, etc., which serve different purposes.

When it comes down to theory and practice, we find that NLTK provides an extensive framework around which one can build systems that work on natural language. This means that it offers a lot more than just an implementation or two; instead, it contains a wide variety of tools for doing various things. For instance, among its capabilities, we can mention models/algorithms development assistance, methods engendering statistical language models trained by n-grams or hidden Markov models (HMMs) creation help, etc., incorporating other systems’ libraries in a unified way.

NLTK offers numerous tools and resources for text classification, sentiment analysis, and machine learning on textual data. For example, it has algorithms such as Naive Bayes, Decision Trees, and Maximum Entropy for document classification. Additionally, NLTK provides techniques for feature extraction, model training, and evaluation that help people create natural language processing models suited to different types of classification problems.

TensorFlow.keras:

Once known as Keras, it is a high-level neural network API that runs on top of TensorFlow. It was developed with the focus of enabling fast experimentation. It is possible to run it on top of TensorFlow, CNTK, or Theano.The provided syntax of Keras is clear and concise, making it easy to build and train models. Additionally, various built-in layers and optimizers offer a wide range of options for beginners in deep learning.

For individuals who are new to this field as well as those who have been around for some time now, Keras remains their top choice when developing machine learning models. This fact can be supported by the number of researchers who use it, including engineers based at Google and Facebook, among other companies.

Benefits of TensorFlow.keras:

- It is easy to use. Keras has a simple syntax for constructing and training models.
- It is powerful. Keras includes many kinds of built-in layers and optimizers, thereby facilitating the beginning of work with deep learning.
- It is flexible. Simple feedforward nets, as well as complex convolutional neural nets, can be built using Keras.
- It is portable. For this reason, Keras runs on top of TensorFlow, CNTK, or Theano, which means that switching between different backends is easy.

Keras:

Keras is a high-level API for creating and training deep learning models. It is written in Python and can be run on top of TensorFlow, CNTK, or Theano. Keras is easy to learn and use. Moreover, there are many facilities to construct deep learning, which makes it the best software for deep learning, e.g.,
A simple and consistent API for building and training models.
A wide range of pre-built layers and models, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and long short-term memory (LSTM) networks.

It can be run on top of many different backends, including TensorFlow, CNTK, and Theano.
There are also various debugging tools along with utilities that help in visualizing models.
Deep learning has become more popular due to Keras, which is easier to understand and use, in addition to containing features beneficial for deep learning. It is also great for beginners because it has many tutorials and resources that can be used for learning purposes.

Use of Keras for Deep Learning:

The building and training of deep learning models are made simple by Keras's easily understood and consistent API.
A wide variety of pre-built layers and models:
When creating your models, you can save a lot of time by using CNNs, RNNs, and LSTMs, among other pre-built layers and models provided by Keras.

Gensim:

Gensim is a library for unsupervised topic modeling and natural language processing, document indexing, and similarity retrieval. It is based on modern statistical machine-learning techniques.Using Python and Cython for performance, Gensim can be used. Large text collections are designed to be processed by Gensim through data streaming and incremental online algorithms, which is different from many other machine learning software packages that only support in-memory processing.

Features of Gensim:

fastText, word2vec, and doc2vec algorithms are implemented in a parallelized way that enables streaming; besides, latent semantic indexing (LSA, LSI, SVD), non-negative matrix factorization (NMF), latent Dirichlet allocation (LDA), tf-idf, and random projections are available under gensim.

Scikit-Learn :

Scikit-Learn is an open-source machine learning library for Python, which is free to use. It offers a range of tools that are efficient in the field of machine learning and statistical modeling such as classification, regression, clustering and dimensionality reduction. NumPy, SciPy, and matplotlib underpin Scikit-learn; it is also meant to work well with other Python libraries.
Scikit-learn is the best choice for machine learning in Python because it is popular due to being simple, efficient and having good documentation. In addition, there exists a big community of users and developers who can provide various materials to assist you understand and utilize this package.

Characteristics of Scikit-Learn:

Simple:
Being straightforward makes understanding its working model easy which consequently leads into efficient usage.

Efficient:
Even though scikit-learn is written in Python, it is optimized for speed by using C and Cython
.
Well-documented:
Scikit-learn has comprehensive documentation that includes tutorials, examples, and API reference documentation.

Large Community:

There’s a huge number of users and developers in the scikit-learn community so you won’t be short on resources for learning how to use it or any other help you might need when working with the library.
Scikit-learn can handle various machine learning tasks such as:
Classification:
It offers several classification algorithms like support vector machines, decision trees, or random forests among others.

Regression:
There are also multiple regression algorithms provided by scikit-learn including linear regression, logistic regression or polynomial regression for instance.

Clustering:
For clustering purposes it’s got you covered with k-means clustering algorithm being one of them but not limited to hierarchical clustering and spectral clustering which are also available through this package too besides others.

Dimensionality Reduction:
Scikit-learn offers different algorithms for dimensionality reduction. These include PCA (Principal Component Analysis) and t-SNE (t-distributed Stochastic Neighbor Embedding).

Re Library :

We will use the re library in Python, which will be the only import required for this notebook. The re library is a powerful tool in Python that allows the programmer to use regular expressions. Regular expression is abbreviated as ‘re’ and it is used to search, manipulate and process text based on specific patterns. When validating user input, extracting information from text or performing complex text transformations among others; if you don’t have any other better option go for this library because it can do all that.

The re library is a crucial tool for handling text data so as to effectively manipulate and process text using regular expressions. It is a utility that is employed in many areas such as web development, data processing, text analysis among others. Becoming an expert in the use of re library will help you a great deal in working with textual information and solving numerous problems associated with text.

Features of Re Library:

Pattern Matching: This library lets you define ‘regular expressions’ which are basically patterns that describe certain things within texts.
Searching and Matching: One can find the first occurrence of a pattern in a string by using re.search() function.
Global Search: The function re.findall() helps to find all occurrences of a pattern within a string and returns a list of matches.

Pattern Compilation: The function re.compile() allows you to compile a regular expression pattern into a regular expression object, which can improve efficiency when the same pattern needs to be used multiple times.
Replacing Text: Its functionality is similar to that of the re.sub() function, which replaces occurrences of a pattern in a string with specified text.
Grouping and Capturing: Regular expression patterns involve using parentheses ( ) that capture specific parts of a match; this is convenient for extracting particular information.

Matplotlib.pyplot:

Matplotlib.pyplot is the library in Python for data visualization. A flexible Python library used to be able to create diverse kinds of visualizations such as lines, bars, scatters, histograms, etc., is called Matplotlib.pyplot. Also, it is used in deep learning for visualizing data and model performance.

Use of Matplotlib.pyplot while working with Deep Learning:

Adaptability:
When dealing with deep learning, one should go for Matplotlib.pyplot since it can help make different types of plots, thereby making it easier to present the model results visually.

Customization:
In addition, customizing these kinds of plots is also possible, so if someone has specific requirements for their project, they may rely on this option too.

Publication-quality output:
There are several formats in which Matplotlib.pyplot plots can be saved. These include PNG, JPEG, and PDF. This feature enables the easy creation of publication-quality plots that can be shared with others.

Seamless integration with the Python ecosystem:
Matplotlib.pyplot works well with other Python libraries, like NumPy and Pandas. Using Matplotlib.pyplot to visualize data stored in NumPy arrays or Pandas Data Frames becomes very easy.

Interactive exploration:
It is possible to make Matplotlib.pyplot plots interactive so that users can explore the data more actively.

Ftfy:

The goal of FTFY is to fix broken Unicode in various ways. Ftfy is designed to take in bad Unicode and output good Unicode so that it can be used in your Unicode-aware code. This is not the same as taking in non-Unicode and outputting Unicode, which is not the goal of ftfy. It is also not meant to shield you from having to write Unicode-aware code. Ftfy helps those who help themselves.

It will be easier if your input is decoded properly and does not have any problems. However, you often cannot control the input; someone else made a mistake, but now it is your problem. In such cases, FTFY will do its best to solve the problem.

Warnings:

The Python warning module is used in programs to issue and deal with warnings. Such a framework allows developers to generate warning messages that can alert users about possible problems or irregularities in their code. This is usually done when a method is marked as deprecated, but it can also be applied to indicate unexpected runtime situations.

Features of the Warning Module:

Controlling how the warnings are displayed:
With the help of this module, developers can specify the format in which warning messages should appear as well as their destination. This means that they can be printed out on the console, stored in a file, or completely ignored, among other options.

Filter Warnings:
The warnings module can filter warnings that developers raise depending on their category, severity, or other criteria. This can help suppress warnings that do not apply to the given context.

Raise Warnings as Exceptions:
Developers also have the capability of using the warnings module to raise warnings as exceptions. This is important in cases where a warning ought to lead to the termination of the program.

Math:

The Python language has a math module that contains built-in mathematical functions.
These functions are used for common mathematical operations such as addition, subtraction, multiplication, and division, as well as trigonometric and logarithmic calculations, etc. Since it provides access to various numeric tools, these modules should be applied when numpy or scipy is insufficient.

Math Functions provided by the Math m\Module Include:

Arithmetic Functions:
It has methods like sum(), subtract(), multiply(), divide(), and power(), which can be applied to both integers and floating-point numbers.
Trigonometrical Functions:
This category contains the methods sine(), cosine(), tangent(), and arccosine(), among others.
Convert degrees to radians:
The degrees() method inverts the value of a number from degrees to radians.
Logarithmic functions:
In the math module, there are functions for logarithmic operations. The logarithm of a number can be calculated by using these functions, for example, log, log10, and log2.