

Project report on

TEXT DETECTION AND ANALYSIS USING TKINTER AND VADER SENTIMENT ANALYSIS

by
(2020-350-020)-Hamza

under supervision of

Dr. Ihtiram Raza Khan
(Assistant Professor)



**computer science engineering with artificial intelligence school
of engineering and science Jamia Hamdard**

Nov.2023

| S_no | Content | page no | Signature |
|------|--|----------|-----------|
| 1 | Declaration and Acknowledgement | 3&4 | |
| 2 | Introduction and Idea behind creating this project | 5 | |
| 3 | Problem statement | 6 | |
| 4 | Purpose | 7 | |
| 5 | Solution | 8 | |
| 6 | MUST SEE Result: through Demonstration clip (Refer to soft copy of this report to play/see.) | 9 | |
| 7 | <u>Flow Chart</u> | 10 | |
| 8 | Flow Chart Explanation | 11 | |
| 9 | Technology used | 13 | |
| 10 | Future scope | 15 | |
| 11 | Limitation | 16 | |
| 12 | Methodology | 17 | |
| 13 | Let's look the code | 21 | |
| 14 | ENTIRE CODE EXPLANATION IN SHORT | 29 | |
| 15 | Conclusion | 32 | |
| 16 | Useful links (code and video link) | 33 | |
| 17 | Thank you!! | 34 | |

DECLARATION

I, **Mr. HAMZA** a student of **Bachelors in Technology Computer Science and Engineering with Artificial Intelligence (B.Tech C.S.E-AI)** hereby declare that the Project/Dissertation entitled “ **TEXT DETECTION AND ANALYSIS USING TKINTER AND VADER SENTIMENT ANALYSIS**” which is being submitted by me to the Department of Computer Science, Jamia Hamdard, New Delhi in partial fulfilment of the requirement for the award of the degree of Bachelors in Technology Computer Science and Engineering artificial intelligence (B.Tech C.S.E-AI), is my original work and has not been submitted anywhere else for the award of any Degree, Diploma, Associateship, Fellowship or other similar title or recognition.

HAMZA

Place: Jamia Hamdard.

ACKNOWLEDGEMENT

I would want to express my gratitude to a number of persons who have encouraged. It allows me to look back and reflect on the support I've had during this process.

I would like to express my gratitude to **Respected Ihtiram Sir**, my mentor, as well as academician, for their invaluable recommendations, innovative criticisms, and support throughout the writing process. I would like to offer my heartfelt gratitude to the entire members Jamia Hamdard faculty.

Introduction and Idea behind creating this project(3in1):



So, I created model which have GUI using Tkinter which is capable to detect text & distinguish into negative and positive after performing Vader sentiment analysis.

Usually, I do online shopping so I faced one **problem** which is reading reviews to decide whether this product holds positive or negative reviews so recently **Amazon** shows positive and negative reviews on left bottom almost of their products so from there I tried to **replicate** it.

Because reading all reviews is too much time consuming and we have less of time, we can you use this system to know the weather in paragraph, sentence is positive/negative or if we want then we can read that.

Problem statement

Build system that can understand the sentiment behind the text (like human does), and show the positive and negative.

Purpose

- it saves our time because to buy any product online we read the product review which consumes lots of time.
- It supports in making quick decision by understanding the sentiment behind the text.
- It also manages our time during busy schedule to do online shopping.

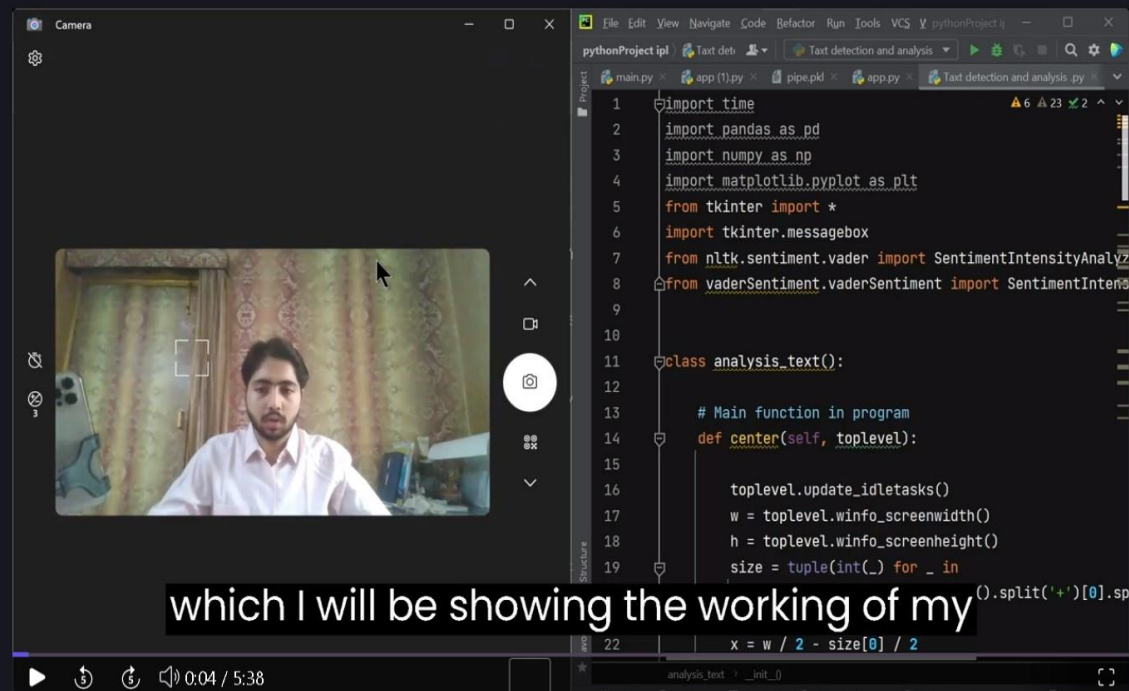
Solution

- I created a software model which is able to tell you the sentiment behind the paragraph, sentence etc.
- Make a graphical user interface to detect the user text so we can conclude that the text positive or negative sentiments.

Result:

DEMONSTRATION CLIP

[CLICK HERE](#) OR ON [IMAGE](#) TO SEE WORKING AND
RESULT OF MY PROJECT.



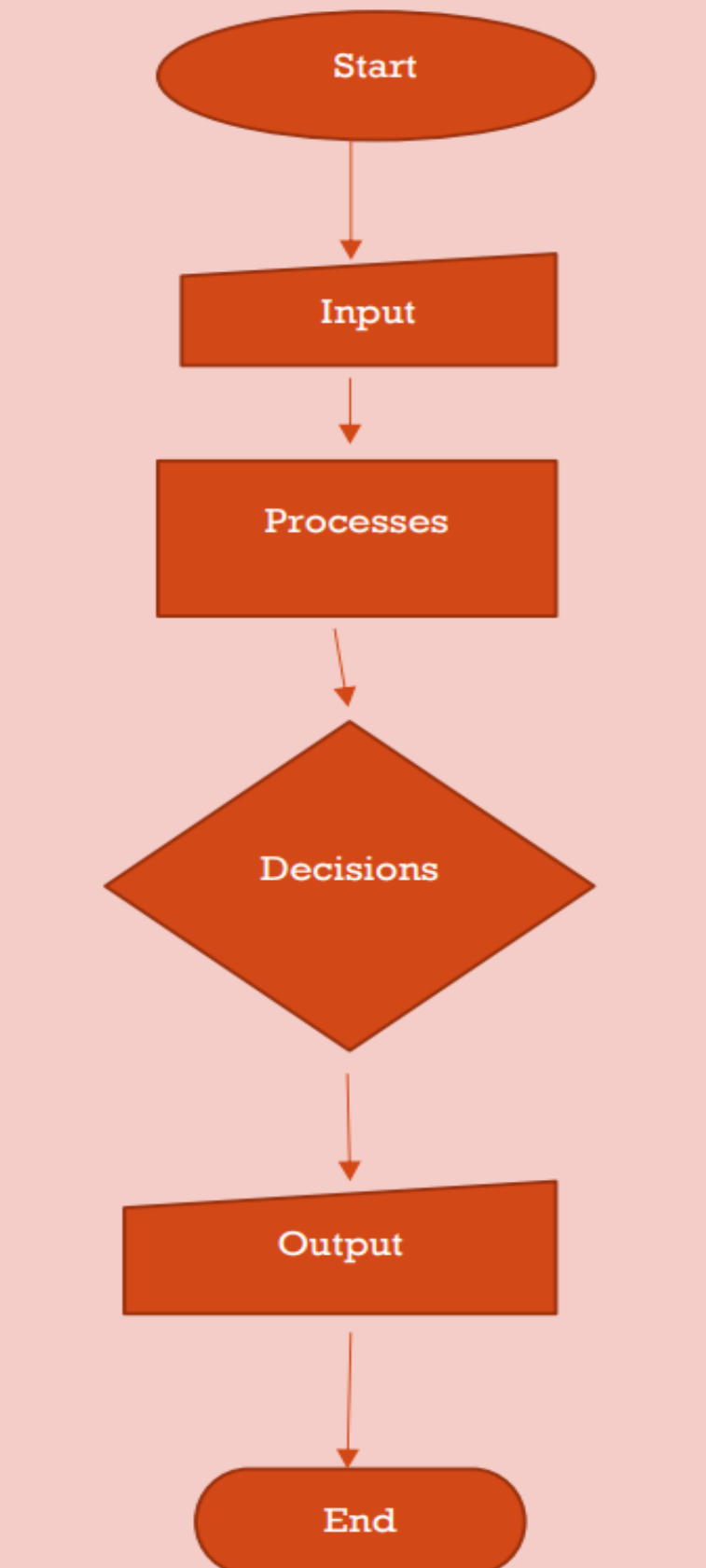
Demonstration project clip

Created by Hamza

[Share this video](#)

[Report video](#)

Flow Chart



Flow Chart Explanation

■Start:

Program begins by importing necessary libraries and modules.

■Class analysis_text:

Initialization (__init__):

Creates the main window and initializes the GUI elements.

Sets up labels and input fields.

Binds events for text editing and analysis. center

Method:

Centers the main window on the screen. callback

Method:

Triggers a prompt to confirm before closing the application. setResult

Method:

Updates labels based on the sentiment analysis results. runAnalysis

Method:

Gathers text input, performs sentiment analysis, and updates labels accordingly.

editedText Method:

Updates the typedText label with the entered text. runByEnter

Method:

Triggers the sentiment analysis method when the 'Enter' key is pressed.

▪Driver Code:

Creates an instance of the analysis_text class (myanalysis).

Enters the Tkinter event loop (mainloop()), allowing user interaction with the GUI.

The flowchart would represent the sequence of steps and how different methods interact with each other in response to user actions (text entry and 'Enter' key press). Each method or function execution can lead to other methods or event triggers, demonstrating the program's workflow.

Technology used

- **Python programming** is extensively used in text detection and sentiment analysis due to its rich set of libraries and tools that facilitate natural language processing (NLP) tasks.
- **Google Collaboratory** in which, I done training of model because it allows you to write codes in cells and run them independently and check the errors and output the control next to each cell.
- **VADER (Valence Aware Dictionary and sEntiment Reasoner)** is a lexicon and rule-based sentiment analysis tool designed to analyze sentiments in text. It's particularly useful for social media text, news articles, customer reviews, and various other forms of online content due to its ability to handle the nuances of informal language and emoticons.
- **PyCharm** is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development. Done coding creating user interface.
- **Tkinter** is a standard GUI (Graphical User Interface) toolkit in Python used for developing desktop applications. When it comes to text detection and sentiment analysis, Tkinter can be

utilized to create a user interface that interacts with text detection algorithms and sentiment analysis tools.

Future scope

- ✚ Deploying some good website like telecom/ e-commerce.
- ✚ Integrating with applications.
- ✚ Adding more functionality(more options in user interface and showing the reason of result)

Limitation

Sarcasm and Irony:

Detecting sarcasm and irony in text can be challenging for sentiment analysis algorithms. Texts often contain sarcastic or ironic comments that can convey the opposite sentiment to the actual meaning of words.

Tone and Emotion Detection:

Sentiment analysis tools might struggle to detect the emotional tone and intention behind a text, especially in cases where there's a mixed sentiment or complex emotional expressions.

Methodology

One can detect an image, speech, can even detect an object through Python. For now, we will detect whether the text from the user gives a positive feeling or negative feeling by classifying the text as positive, negative, or neutral. To implement it I used, Vader sentiment analysis and Tkinter are used. Tkinter is a standard GUI library for creating the GUI application.

Required Installations in Anaconda:

tkinter: This module is used for creating a GUI application. This module generally comes pre-installed with Python but to install it externally type the below command in the terminal.

Using conda command.

```
conda install -c anaconda tk
```

nltk: This module is used for making computers understand the natural language. To install it type the below command in the terminal.

Using conda.

Using pip. pip

```
install nltk
```

numpy: This module is the fundamental package for scientific computing with Python. To install it type the below command in the terminal.

Using conda.

```
pip install numpy
```

pandas: This module is used for data analysis. It provides highly optimized performance with back-end source code is purely written in C or Python. To install it type the below command in the terminal. Using conda

```
pip install pandas
```

matplotlib: This module is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays. To install it type the below command in the terminal.

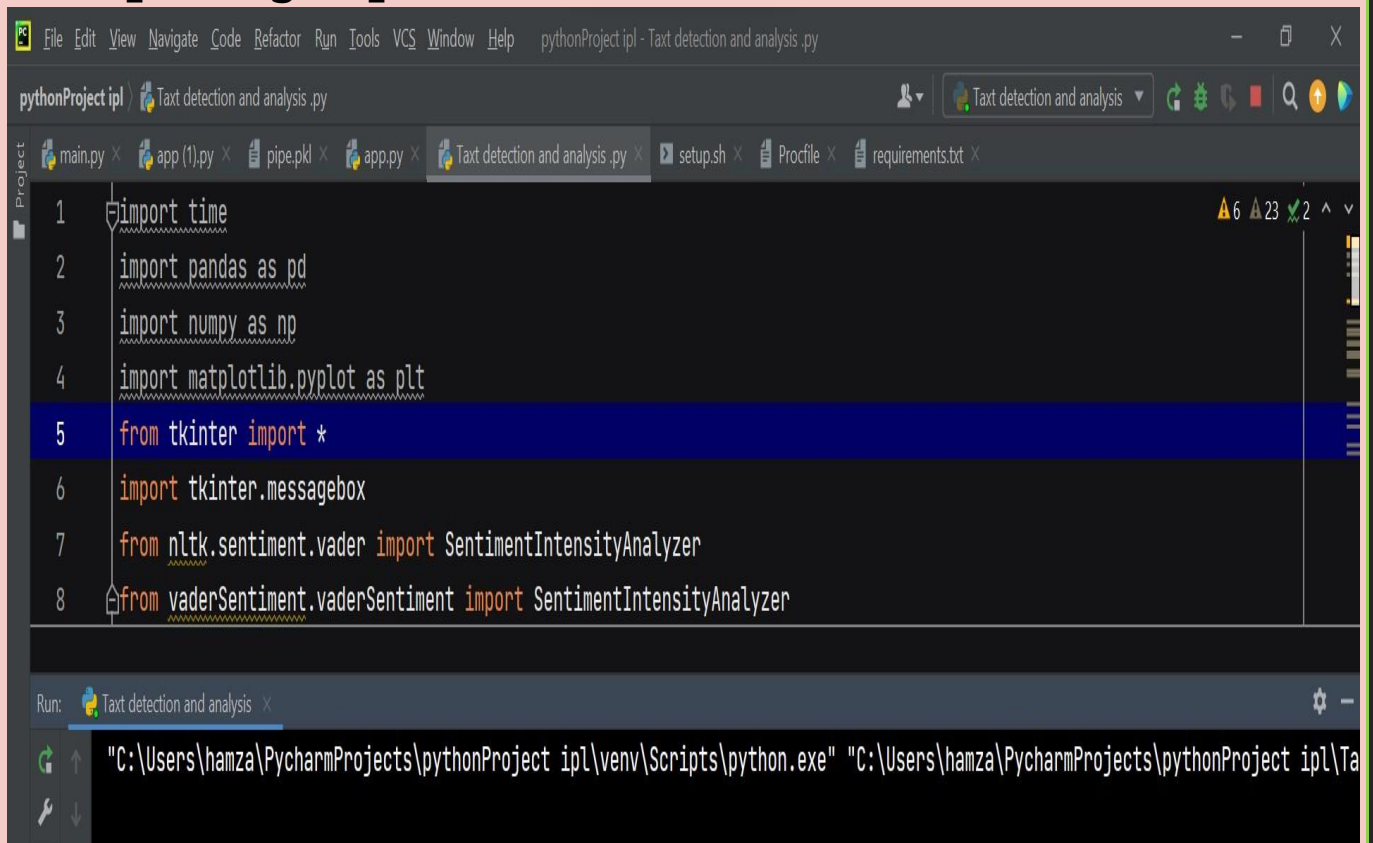
Using conda.

```
pip install matplotlib
```

VADER Sentiment Analysis

VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. VADER uses a combination of A sentiment lexicon is a list of lexical features (e.g., words) which are generally labeled according to their semantic orientation as either positive or negative. VADER not only tells about the Positivity and Negativity score but also tells us about how positive or negative a sentiment is.

Importing dependencies



The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The title bar indicates the project is 'pythonProject ip1' and the current file is 'Text detection and analysis .py'. The toolbar shows icons for running, debugging, and other actions. The file explorer on the left shows the project structure with files like main.py, app (1).py, pipe.pkl, app.py, Text detection and analysis .py, setup.sh, Profile, and requirements.txt. The main editor window displays the following Python code:

```
1 import time
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
5 from tkinter import *
6 import tkinter.messagebox
7 from nltk.sentiment.vader import SentimentIntensityAnalyzer
8 from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
```

Below the editor is a terminal window titled 'Run: Text detection and analysis'. It shows the command being executed: `"C:\Users\hamza\PycharmProjects\pythonProject ip1\venv\Scripts\python.exe" "C:\Users\hamza\PycharmProjects\pythonProject ip1\Ta`

time module in your Python scripts to work with time. You can do actions such as retrieving the current time, waiting during code execution, and measuring the efficiency of your code.

pandas for data analysis.

numpy perform array processing which allows you to efficiently operate on multiple elements of an array at once; access high performance computing solutions

matplotlib to displays the plot using the `show ()` function

tkinter represents all the functions and built-in modules in the tkinter library. By importing all the functions and methods, we can use the inbuilt

functions or methods in a particular application without importing them implicitly.

messagebox used to display message boxes in your applications. This module provides a number of functions that you can use to display an appropriate message.

nlTK It is a really powerful tool to preprocess text data for further analysis like with ML models for instance. It helps convert text into numbers, which the model can then easily work with.

SentimentIntensityAnalyzer takes in a string and returns a dictionary of scores in each of categories: negative. neutral. positive.

LET'S LOOK THE CODE

```
class analysis_text():  
  
    # Main function in program  
    def center(self, toplevel):  
  
        toplevel.update_idletasks()  
        w = toplevel.winfo_screenwidth()  
        h = toplevel.winfo_screenheight()  
        size = tuple(int(_) for _ in  
                      toplevel.geometry().split('+')[0].split('x'))  
  
        x = w / 2 - size[0] / 2  
        y = h / 2 - size[1] / 2  
        toplevel.geometry("%dx%d+%d+%d" % (size + (x, y)))
```

here we are creating the function for height and width.

different functions like

Geometry decides the size, position and some other attributes of the screen layout we are going to create. **Split** method splits a string into a list.

screen width and height.

```
26 def callback(self):
27     if tkinter.messagebox.askokcancel("Quit",
28                                         "Do you want to leave?"):
29         self.main.destroy()
30
31 def setResult(self, type, res):
32
33     # calculated comments in vader analysis
34     if (type == "neg"):
35         self.negativeLabel.configure(text=
36                                     "you typed negative comment : "
37                                     + str(res) + " % \n")
38     elif (type == "neu"):
39         self.neutralLabel.configure(text=
40                                     "you typed comment : "
41                                     + str(res) + " % \n")
42     elif (type == "pos"):
43         self.positiveLabel.configure(text
44                                     ="you typed positive comment: "
45                                     + str(res) + " % \n")
46
```

analysis_text > setResult() > elif (type == "neu")

38:30 CRLF UTF-8 4 spaces Python 3.9 (pythonProject ip)

08:12 PM

Creating another function for call back in which using tkinter, using different functions like ask to cancel **Destroy** it terminates the mainloop process and destroys all the widgets inside the window. configure function used to access an object's attributes after its initialisation. and making quit option and making sentence for positive negative and natural statements.

```
app (1).py × pipe.pkl × app.py × Text detection and analysis .py × setup.sh × Procfile × require
def runAnalysis(self):
    sentences = []
    sentences.append(self.line.get())
    sid = SentimentIntensityAnalyzer()

    for sentence in sentences:

        # print(sentence)
        ss = sid.polarity_scores(sentence)

        if ss['compound'] ≥ 0.05:
            self.normalLabel.configure(text=
                                        " you typed positive statement: ")

        elif ss['compound'] ≤ - 0.05:
            self.normalLabel.configure(text=
                                        " you typed negative statement")

        else:
            self.normalLabel.configure(text=
                                        " you normal typed statement: ")

        for k in sorted(ss):
            self.setResult(k, ss[k])
    print()
```

This function is to run the analyser which is main part of this project.

Priority score function Return a float for sentiment strength based on the input text.

Positive values are positive valence, negative value are negative valence.

VADER's SentimentIntensityAnalyzer() takes in a string and returns a dictionary of scores in each of four categories: negative, neutral,

```
def editedText(self, event):  
    self.typedText.configure(text=self.line.get() + event.char)
```

Responds to user input or events (like keyboard typing) and updates the displayed text dynamically in a graphical user interface (GUI).

```
def runByEnter(self, event):  
    self.runAnalysis()
```

used in a GUI application where runByEnter is a callback function that's activated when the 'Enter' key is pressed, and it, in turn, triggers the runAnalysis process in the application.

```
def __init__(self):  
    # Create main window  
    self.main = Tk()  
    self.main.title("Text Detector system")  
    self.main.geometry("600x600")  
    self.main.resizable(width=FALSE, height=FALSE)  
    self.main.protocol("WM_DELETE_WINDOW", self.callback)  
    self.main.focus()  
    self.center(self.main)
```

initializing a GUI window and configures its properties and behavior when the program starts.

Giving the name to GUI and size.

```
# addition item on window
self.label1 = Label(text="type a text here :")
self.label1.pack()
```

Adding more functionalities option to the GUI.

To take user input.

```
# Add a hidden button Enter
self.line = Entry(self.main, width=70)
self.line.pack()
```

It will not be visible on screen but when you press you will get desired function.

```
self.textLabel = Label(text="\n",
                        font=("Helvetica", 15))
self.textLabel.pack()
self.typedText = Label(text="",
                        fg="blue",
                        font=("Helvetica", 20))
self.typedText.pack()
```

This code sets up two Label widgets: one displaying a newline character (possibly for formatting or spacing purposes) and another initially empty label, styled with a larger font size and blue text color. Both labels are added to the GUI window for display using the pack () method.

```
app (1).py × pipe.pkl × app.py × Text detection and analysis .py × setup.sh × P

self.line.bind("<Key>", self.editedText)
self.line.bind("<Return>", self.runByEnter)

self.result = Label(text="\n",
                    font=("Helvetica", 15))
self.result.pack()
self.negativeLabel = Label(text="",
                          fg="red",
                          font=("Helvetica", 20))
self.negativeLabel.pack()
self.neutralLabel = Label(text="",
                          font=("Helvetica", 20))
self.neutralLabel.pack()
```

`self.line.bind("<Key>", self.editedText)`: This line of code is binding the `editedText` method or function to an event that occurs when a key is pressed within the `self.line` widget. Whenever a key is pressed, the `editedText` function will be called.

`self.line.bind("<Return>", self.runByEnter)`: This line is binding the `runByEnter` method or function to an event that occurs when the "Return" or "Enter" key is pressed within the `self.line` widget. This function will be executed when the "Enter" key is pressed.

`self.result`, `self.negativeLabel`, and `self.neutralLabel` are instances of the `Label` class in Tkinter, representing labels in the GUI.

`self.result` is a label set with initial text `"\n"` (which represents a new line) and a font size of 15. It's then added to the GUI using the `pack()` method.

`self.negativeLabel` and `self.neutralLabel` are labels left empty in terms of initial text. `self.negativeLabel` is configured to display text in red with a larger font size (size 20) and is added to the GUI using `pack()`. `self.neutralLabel` is also added to the GUI but with a default black color and a font size of 20.

creating a GUI, capturing keyboard inputs, and displaying different labels for text output or display within the user interface. The `bind()` function is used to associate key events with specific functions, and the `Label` class is used to create and configure text labels that are then added to the GUI for display.

```
self.positiveLabel = Label(text="",
                            fg="green",
                            font=("Helvetica", 20))
self.positiveLabel.pack()
self.normalLabel = Label(text="",
                          fg="red",
                          font=("Helvetica", 20))
self.normalLabel.pack()
```

These labels are intended to display different types of information within the GUI window, such as neutral, positive, or normal messages. They are configured to have different text colors but are initially empty. The specific usage and content that will be displayed in these labels would likely be defined and updated elsewhere in the code when the program runs.

```
# Driver code
myanalysis = analysis_text()
mainloop()
```

Is used to start the event loop, which listens for events (such as keypresses or mouse clicks) and responds to these events by calling the associated functions or

methods. This method continues to run until the window is closed by the user or explicitly terminated, keeping the GUI responsive.

ENTIRE CODE EXPLANATION IN SHORT

THIS CODE DEFINES A PYTHON CLASS ANALYSIS_TEXT THAT IMPLEMENTS A MODERATE TEXT ANALYSIS SYSTEM USING THE **TKINTER** LIBRARY FOR CREATING A GRAPHICAL USER INTERFACE (GUI) AND THE **NLTK** LIBRARY FOR SENTIMENT ANALYSIS.

LOOK THE COLD PARTS

IMPORTING LIBRARIES:

TIME, PANDAS, NUMPY, MATPLOTLIB.PYLOT: THESE ARE COMMON LIBRARIES FOR HANDLING TIME-RELATED FUNCTIONS, DATA MANIPULATION, NUMERICAL OPERATIONS, AND PLOTTING GRAPHS.

FROM TKINTER IMPORT *: IMPORTS THE ENTIRE TKINTER LIBRARY FOR GUI DEVELOPMENT.

TKINTER.MESSAGEBOX: SPECIFICALLY IMPORTS THE MESSAGEBOX MODULE FROM TKINTER FOR DISPLAYING MESSAGE BOXES.

FROM NLTK.SENTIMENT.VADER IMPORT SENTIMENTINTENSITYANALYZER: IMPORTS THE SENTIMENTINTENSITYANALYZER CLASS FROM NLTK'S VADER (VALENCE AWARE DICTIONARY AND SENTIMENT REASONER) MODULE FOR SENTIMENT ANALYSIS.

FROM VADERSENTIMENT.VADERSENTIMENT IMPORT SENTIMENTINTENSITYANALYZER:

CLASS ANALYSIS_TEXT:

THIS CLASS CONTAINS METHODS THAT DEFINE THE BEHAVIOR OF THE TEXT ANALYSIS GUI.

CENTER: DEFINES A METHOD TO CENTER THE WINDOW ON THE SCREEN.

CALLBACK: HANDLES THE CLOSING EVENT OF THE WINDOW AND ASKS FOR CONFIRMATION BEFORE QUITTING.

SETRESULT: UPDATES LABELS BASED ON THE SENTIMENT ANALYSIS RESULTS OBTAINED.

RUNANALYSIS: ANALYZES THE TEXT ENTERED IN THE GUI BY COMPUTING ITS SENTIMENT POLARITY USING THE VADER SENTIMENT ANALYSIS TOOL FROM NLTK. IT UPDATES THE GUI LABELS BASED ON THE SENTIMENT ANALYSIS RESULTS.

EDITEDTEXT: UPDATES THE TYPEDTEXT LABEL IN REAL-TIME AS THE USER TYPES INTO THE INPUT FIELD.

RUNBYENTER: TRIGGERS THE SENTIMENT ANALYSIS WHEN THE 'ENTER' KEY IS PRESSED IN THE INPUT FIELD.

__INIT__: INITIALIZES THE MAIN GUI WINDOW WITH LABELS, AN INPUT FIELD, AND

SETS UP THE EVENT BINDINGS FOR TEXT EDITING AND ANALYSIS.

DRIVER CODE:

CREATES AN INSTANCE OF THE ANALYSIS_TEXT CLASS.

ENTERS THE TKINTER EVENT LOOP TO HANDLE GUI EVENTS.

THIS CODE ESSENTIALLY CREATES A INTERMEDIATE **GUI** WHERE USERS CAN INPUT TEXT, AND UPON PRESSING 'ENTER,' THE PROGRAM PERFORMS **SENTIMENT** ANALYSIS ON THE TEXT AND DISPLAYS THE ANALYZED RESULTS (POSITIVE, NEGATIVE, OR NEUTRAL SENTIMENT) WITHIN THE **GUI**. THE SENTIMENT ANALYSIS ITSELF IS BASED ON THE **VADER** TOOL PROVIDED BY **NLTK**.

conclusion

The project on Text Detection and Analysis utilizing Tkinter and VADER sentiment analysis offers a comprehensive exploration into text-based sentiment analysis through a user-friendly graphical interface. The integration of Tkinter, a powerful GUI toolkit, and VADER sentiment analysis, a lexicon and rule-based sentiment analysis tool, provides a functional platform for users to input text, analyze sentiments, and receive immediate feedback regarding the sentiment of their text inputs.

In conclusion, this project serves as a foundational demonstration of the amalgamation of a user interface created with Tkinter and the capabilities of VADER sentiment analysis. It provides a solid framework for further enhancements and the potential for integration into various applications, including social media analysis, customer feedback processing, and broader natural language processing tasks. Continuing advancements in sentiment analysis methodologies and user interface design will further refine and expand the capabilities demonstrated in this project.

Useful links

link to watch live working of this project by me

<https://clipchamp.com/watch/ZOIcx6l6Lgi>

Link to download the clip demonstrating the result.

[https://drive.google.com/file/d/1Lz0ymUQxa0hYQrnbbwr3OA21oAznfq1S/view?usp=drive link](https://drive.google.com/file/d/1Lz0ymUQxa0hYQrnbbwr3OA21oAznfq1S/view?usp=drive_link)

