

Name :- Aishwarya Verma  
Email :- aishwaryaverma56@gmail.com

Q. Develop a chatbot equipped with sentiment analysis capabilities. The chatbot will analyze the sentiment of the user's input. The sentiment analysis component will determine whether the user's message expresses a positive, negative, or neutral sentiment.

This project combines natural language processing (NLP) techniques, machine learning algorithms To Find the Sentiment Of User

Solution:

### Natural Language Processing (NLP)

Natural Language Processing is a branch of Artificial Intelligence that facilitates communication between computers and humans by enabling machines to understand , interpret and generate human-like language. It involves techniques like tokenization , part-of-speech tagging and sentiment analysis to process and analyse text data.

NLP plays a crucial role in application such as chat-bots , language translation ,and sentient analysis enhancing the capabilities of systems to comprehend and respond to human language in a meaningful way.

### Sentimental Analysis

Sentiment analysis is a branch of natural language processing (NLP) that identifies the sentiment present in a text and categorises it as either positive, negative, or neutral. Thanks to this clever technique, machines can now comprehend and interpret human feelings and opinions in a variety of settings, including social media, consumer evaluations, and feedback. Sentiment analysis, which gauges the emotional tone associated with textual data, is applied across sectors and offers significant insights for decision-making, brand management, and improving user experiences.

### Sentiment Analysis Chat-bot

The TextBlob library is used by the sentiment analysis chatbot to determine the sentiment of user input. After receiving a text input, the `analyze_sentiment` function determines the sentiment of the text and returns the sentiment label.

### Code

```
from tkinter import Tk, Text, Scrollbar, END, Frame, ttk
from textblob import TextBlob
import nltk

try:
    nltk.data.find('tokenizers/punkt')
except LookupError:
    nltk.download('punkt')
try:
    from textblob.download_corpora import main
    main()
except nltk.exceptions.ContentRetrievalError:
    print("Error: Could not download the necessary corpora for the textblob library.")

class SentimentAI:
```

```

def __init__(self, master):
    self.master = master
    master.title("Sentiment Analysis Chatbot")

    self.frame = Frame(master, padx=20, pady=20)
    self.frame.pack()

    self.label = ttk.Label(self.frame, text="Enter Text:")
    self.label.grid(row=0, column=0, sticky='w')

    self.entry = ttk.Entry(self.frame, width=40)
    self.entry.grid(row=0, column=1, padx=5, pady=5)

    self.button = ttk.Button(self.frame, text="Analyze", command=self.analyze)
    self.button.grid(row=0, column=2, pady=10)

    self.reset_button = ttk.Button(self.frame, text="Reset", command=self.reset)
    self.reset_button.grid(row=0, column=3, padx=5, pady=10)

    self.output_text = Text(self.frame, wrap='word', height=10, width=50, state='normal')
    self.output_text.grid(row=1, column=0, columnspan=4, padx=5, pady=5)

    self.scrollbar = Scrollbar(self.frame, command=self.output_text.yview)
    self.scrollbar.grid(row=1, column=4, sticky='ns')

    self.output_text.config(yscrollcommand=self.scrollbar.set)

def fade_in(self, widget, alpha=0):
    widget.attributes("-alpha", alpha)
    alpha += 0.1
    if alpha <= 1.0:
        self.master.after(50, self.fade_in, widget, alpha)

def display_analysis_result(self, output_message):
    self.output_text.config(state='normal')
    self.output_text.delete(1.0, END)
    self.output_text.insert(END, output_message + '\n')
    self.output_text.config(state='disabled')

def reset(self):
    self.entry.delete(0, 'end')
    self.output_text.config(state='normal')
    self.output_text.delete(1.0, END)
    self.output_text.config(state='disabled')

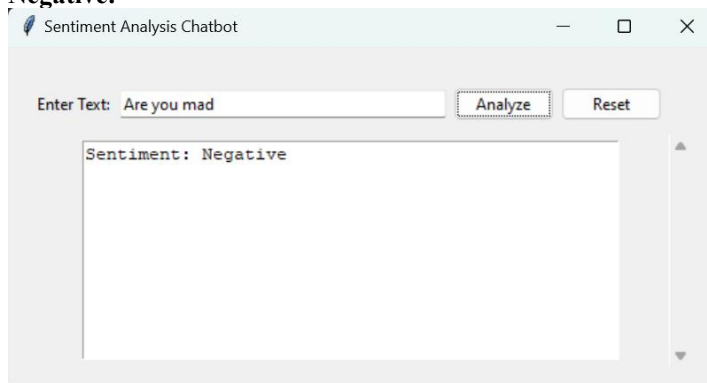
def analyze(self):
    text = self.entry.get()
    if text:
        analysis = TextBlob(text)
        polarity, subjectivity = analysis.sentiment.polarity, analysis.sentiment.subjectivity
        sentiment = 'Positive' if polarity > 0 else 'Negative' if polarity < 0 else 'Neutral'
        output_message = f"Sentiment: {sentiment}"
        self.display_analysis_result(output_message)

if __name__ == "__main__":
    root = Tk()
    app = SentimentAI(root)
    app.fade_in(root)
    root.mainloop()

```

## Result:

### **Negative:**



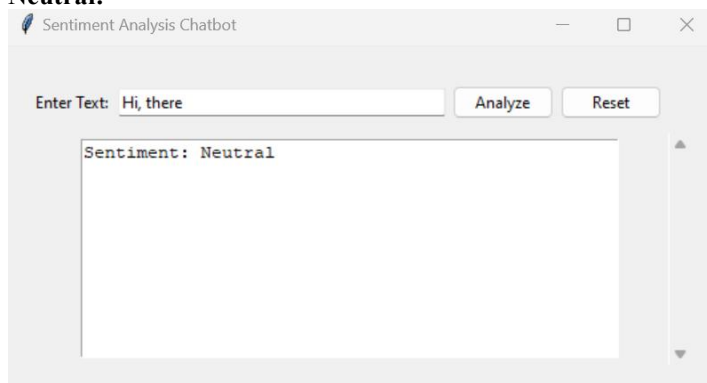
Sentiment Analysis Chatbot

Enter Text:

Sentiment: Negative

The screenshot shows a web application window titled "Sentiment Analysis Chatbot". It features a text input field containing the phrase "Are you mad", followed by "Analyze" and "Reset" buttons. Below the input, a text area displays the result "Sentiment: Negative".

### **Neutral:**



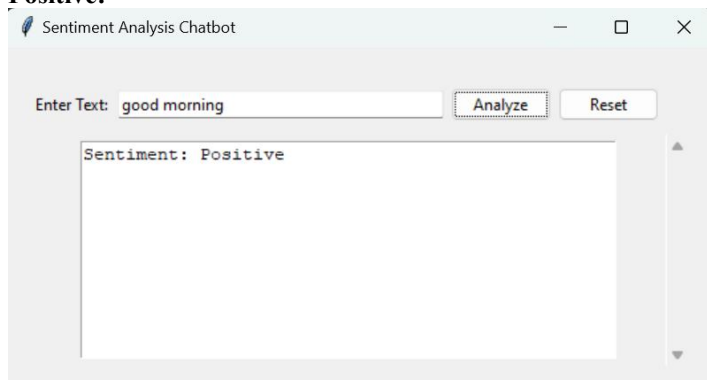
Sentiment Analysis Chatbot

Enter Text:

Sentiment: Neutral

The screenshot shows the same web application window. The text input field now contains "Hi, there", and the result displayed in the text area is "Sentiment: Neutral".

### **Positive:**



Sentiment Analysis Chatbot

Enter Text:

Sentiment: Positive

The screenshot shows the same web application window. The text input field now contains "good morning", and the result displayed in the text area is "Sentiment: Positive".