Name :- Kinshuk
Email :- kinshuksinha095@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

Answer:

Processing of Natural Language (NLP)

Artificial intelligence's field of natural language processing enables machines to understand, interpret, and produce language that is similar to human speech, hence promoting human-computer communication. It uses techniques like tokenisation, sentiment analysis, and part-of-speech tagging to handle and interpret text data.
Because natural language processing (NLP) enables algorithms to comprehend and respond meaningfully to human language, it is critical to the operation of applications such as chatbots, language translation, and intelligent analysis.

Sentimental Analysis

Sentiment analysis, a branch of natural language processing (NLP), classifies texts as good, negative, or neutral based on the sentiment that is found within them. Thanks to this clever technique, robots can now comprehend and interpret human emotions and opinions in a variety of settings, including social media, customer evaluations, and feedback. One popular method for evaluating the emotional tone of textual data is sentiment analysis. It offers insightful information for managing brands, making decisions, and improving user experiences.

Chatbot for Sentiment Analysis

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

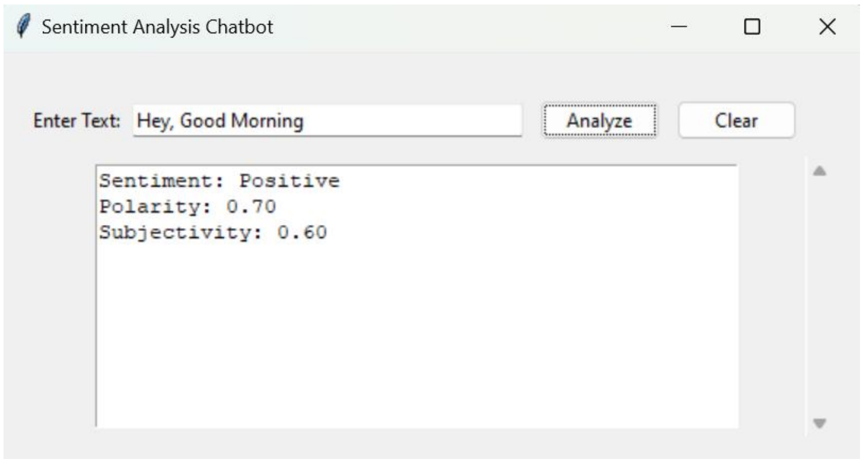
Code

```
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 Exception:
    print("Error: Could not download the necessary corpora for the textblob library.")
```

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

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

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

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

        self.analyze_button = ttk.Button(self.main_frame, text="Analyze", command=self.analyze_sentiment)
        self.analyze_button.grid(row=0, column=2, padx=5, pady=10)

        self.clear_button = ttk.Button(self.main_frame, text="Clear", command=self.clear_text)
        self.clear_button.grid(row=0, column=3, padx=5, pady=10)

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

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

        self.result_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_result(self, message):
        self.result_text.config(state='normal')
        self.result_text.delete(1.0, END)
        self.result_text.insert(END, message + '\n')
        self.result_text.config(state='disabled')

    def clear_text(self):
        self.text_entry.delete(0, 'end')
        self.result_text.config(state='normal')
        self.result_text.delete(1.0, END)
        self.result_text.config(state='disabled')

    def analyze_sentiment(self):
        text = self.text_entry.get()
        if text:
            analysis = TextBlob(text)
            polarity = analysis.sentiment.polarity
            subjectivity = analysis.sentiment.subjectivity
            sentiment = 'Positive' if polarity > 0 else 'Negative' if polarity < 0 else 'Neutral'
            result_message = f"Sentiment: {sentiment}\nPolarity: {polarity:.2f}\nSubjectivity: {subjectivity:.2f}"
            self.display_result(result_message)

if __name__ == "__main__":
    root = Tk()
    root.attributes("-alpha", 0)
    app = SentimentAnalysis(root)
    app.fade_in(root)
    root.mainloop()
```

Result:

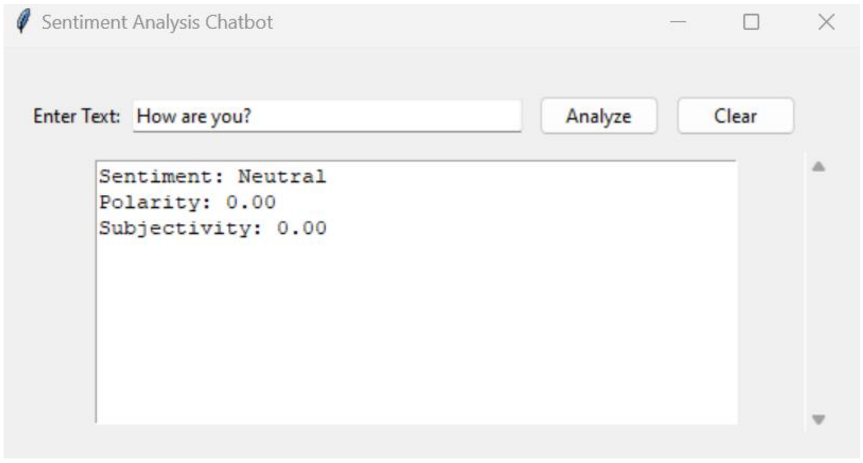
Positive:



Hey, Good Morning

```
Sentiment: Positive
Polarity: 0.70
Subjectivity: 0.60
```

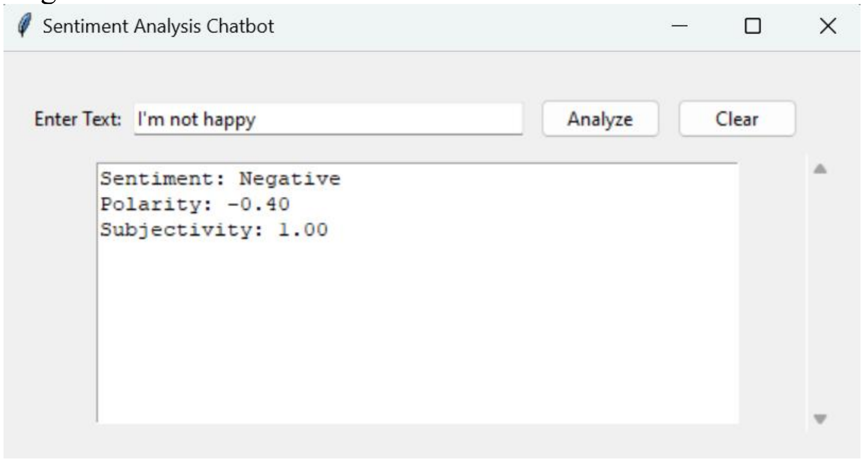
Neutral:



How are you?

```
Sentiment: Neutral
Polarity: 0.00
Subjectivity: 0.00
```

Negative:



I'm not happy

```
Sentiment: Negative
Polarity: -0.40
Subjectivity: 1.00
```