Programming For AI

AI TEXT SENTIMENTAL ANALYZER

Group Members:
Eman Fatima
Arfa Rehman

CONTENTS

INTRODUCTION

- Sentiment Analysis is part of AI
- It analyzes emotions from text
- Used in reviews, feedback, comments
- Our project works on text input
- Shows Positive, Negative or Neutral
- Uses rule-based AI logic

PROBLEM STATEMENT

Manual sentiment analysis is slow

Beginners find AI complex

Large text data is hard to analyze

Need simple AI solution GUI based system is required

STEP 1: Importing Required Libraries

```
from tkinter import *
from tkinter import messagebox
```

Explanation:
tkinter is a built-in Python library used to create a Graphical User Interface (GUI).
messagebox is used to display error or warning pop-up messages.

STEP 2: Defining Positive and Negative Word Lists:

```
positive_words = ["good", "great", "excellent", "happy", "love", "amazing",
"nice", "awesome"]

negative_words = ["bad", "worst", "sad", "hate", "angry", "poor",
"terrible", "boring"]
```

Explanation:

These lists act as the knowledge base of the AI system.
The program uses these words to identify emotions in the input text.
Positive words increase the positive score, and negative words increase the negative score.

STEP 3: Defining the Sentiment Analysis Function:

```
def analyze_sentiment():
```

Explanation:
This function contains the main AI logic of the project.
It runs when the user clicks the “Analyze Sentiment” button.
This is an example of event- driven programming.

STEP 4: Getting User Input from GUI:

```
text = text_input.get("1.0", END)
text = text.strip().lower()
```

Explanation:

The text entered by the user is read from the text box.
strip() removes extra spaces.
lower() converts all text to lowercase so word matching becomes easier.
➞ This step is called texT preprocessing.

STEP 5: Checking for Empty Input:

```
If text == "":
    messagebox.showerror("Error", "Please
enter some text")
    return
```

Explanation:
If the user does not enter any text, an error message is shown.
The function stops execution using return.
➞ This improves **user experience and error handling**

STEP 6: Initializing Counters:

```
positive_count = 0
negative_count = 0
```

Explanation:
These variables store the number of positive and negative words found in the text.
Both counters start from zero.
➞ This is used for **tracking sentiment frequency**.

STEP 7: Tokenization (Splitting Text into Words):

words = text.split()

Explanation:
The sentence is split into individual words.
Example:
"this project is good"
❖ ["this", "project", "is", "good"]
➞ Tokenization is a **basic Natural Language Processing (NLP) step**

STEP 8: Looping Through Each Word:

for word in words:

Explanation:
A loop checks each word one by one.
This allows the system to analyze the entire sentence.
➞ Loops help the AI **process text sequentially**.

STEP 9: Rule-Based Word Matching:

```
if word in positive_words:
    positive_count += 1
elif word in negative_words:
    negative_count += 1
```

Explanation:

Each word is compared with predefined positive and negative lists.

Matching words increase the respective counters.

➞ This is **pattern matching using rules**, not machine learning.

STEP 10: Decision Making (AI Reasoning)

```
if positive_count > negative_count:
    result = "Positive 😊"
elif negative_count > positive_count:
    result = "Negative 😞"
else:
    result = "Neutral 😐"
```

Explanation:

Simple **if-else rules** decide the final sentiment.

The sentiment depends on which type of words appear more.

➞ This is the **decision-making component of AI**.

STEP 11: Displaying the Output on GUI

```
result_label.config(
    text= f"Sentiment: {result}\n Positive
Words: {positive_count}\n Negative Words:
{negative_count}"
)
```

Explanation:

The result is shown on the GUI using a label.

It displays:

- Final sentiment
- Number of positive words
- Number of negative words

➞ This step completes **human-computer interaction**.

STEP 12: Creating the GUI Window

```
root = Tk()
root.title("Rule-Based Sentiment
Analyzer")
root.geometry("500x400")
```

Explanation:

Creates the main application window.
Sets the title and window size.

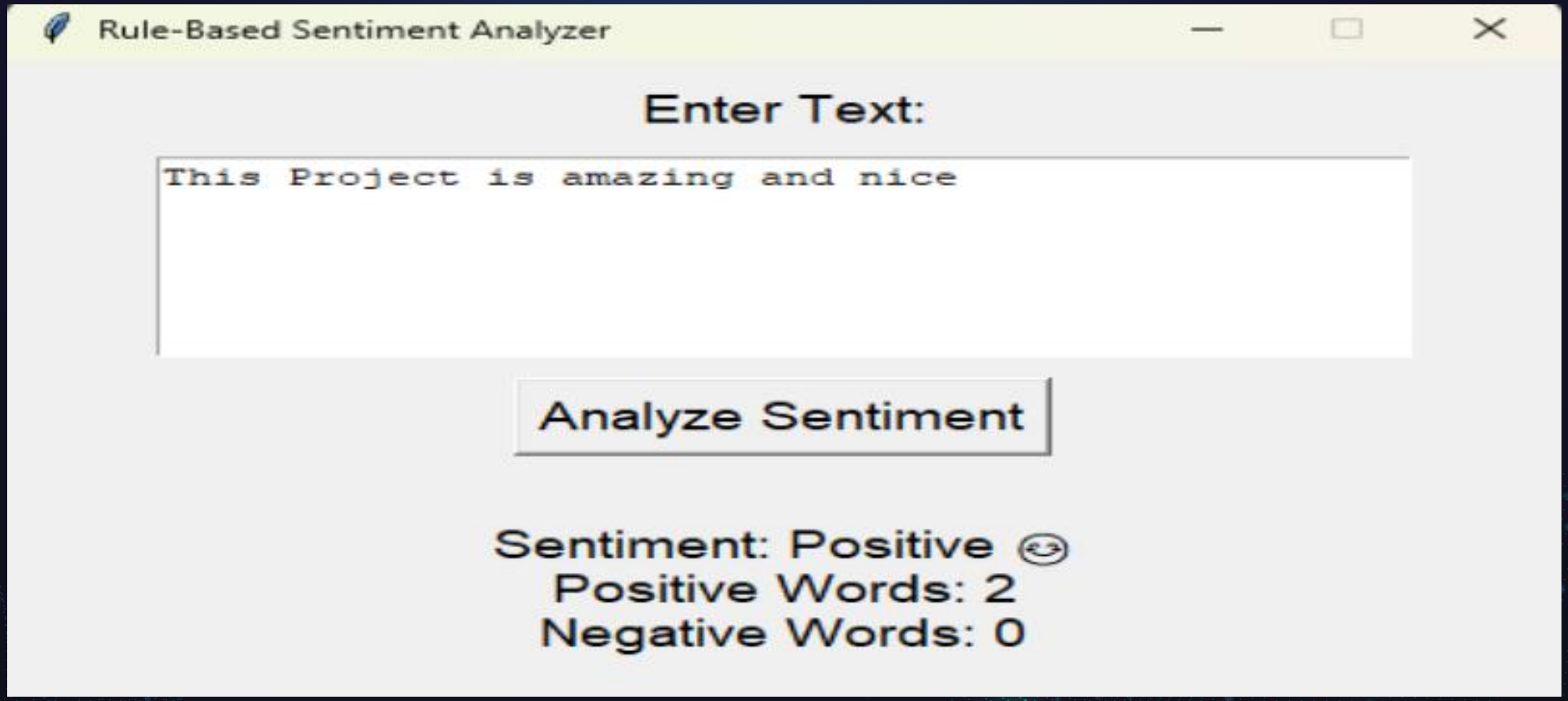
```python
from tkinter import *
from tkinter import messagebox

# ---------- Rule-based word lists ----------
positive_words = [
    "good", "great", "excellent", "happy", "love",
    "amazing", "nice", "awesome", "fantastic"
]

negative_words = [
    "bad", "worst", "sad", "hate", "angry",
    "poor", "terrible", "awful", "boring"
]

# ---------- Sentiment Function ----------
def analyze_sentiment():
    text = text_input.get("1.0", END).strip().lower()

    if text == "":
        messagebox.showerror("Error", "Please enter some text")
        return

```

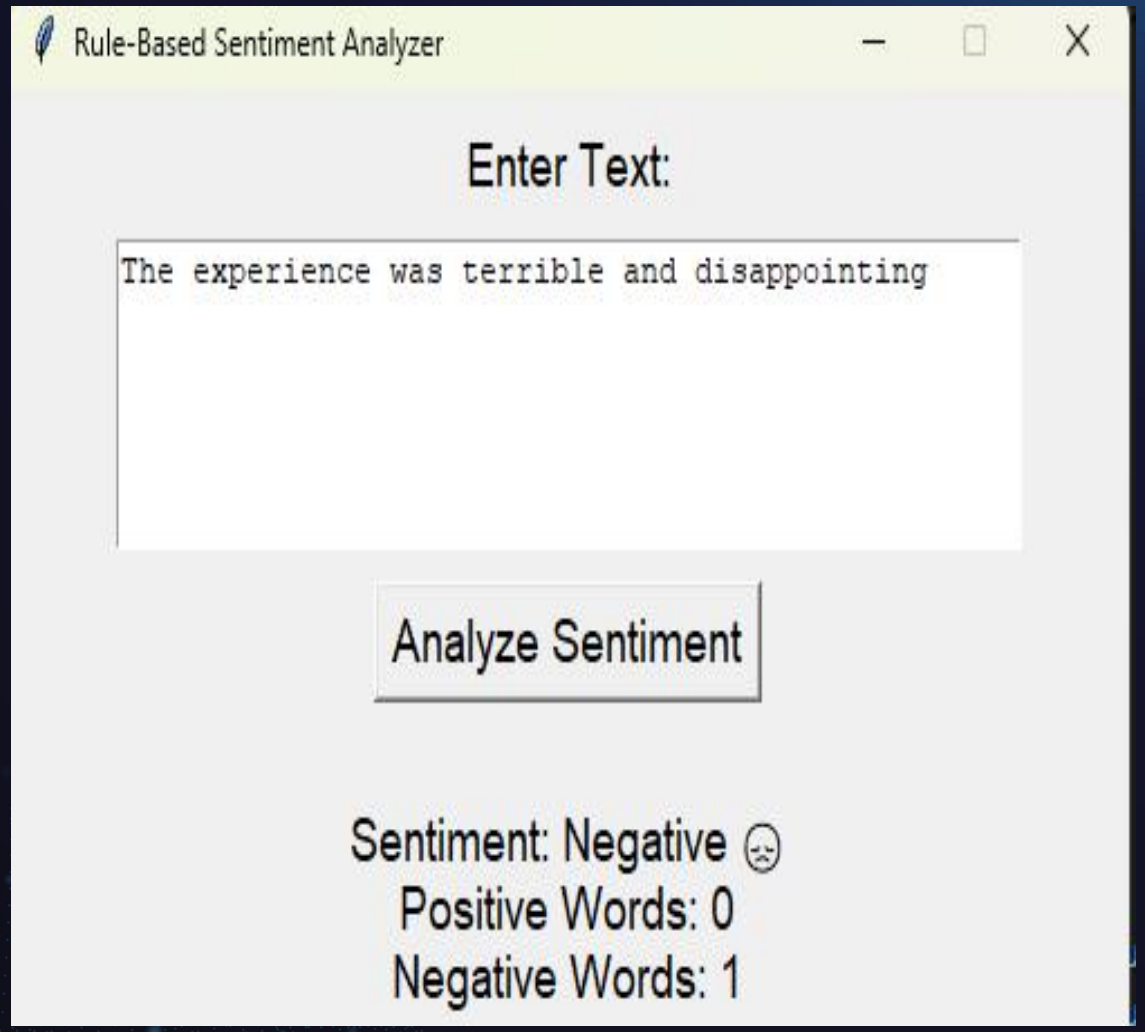
```
    pos_count = 0
    neg_count = 0

    words = text.split()

    for word in words:
        if word in positive_words:
            pos_count += 1
        elif word in negative_words:
            neg_count += 1

    if pos_count > neg_count:
        sentiment = "Positive 😊"
    elif neg_count > pos_count:
        sentiment = "Negative 😞"
    else:
        sentiment = "Neutral 😐"

    result_label.config(
        text=f"Sentiment: {sentiment}\nPositive words: {pos_count}\nNegative words: {neg_count}"
```

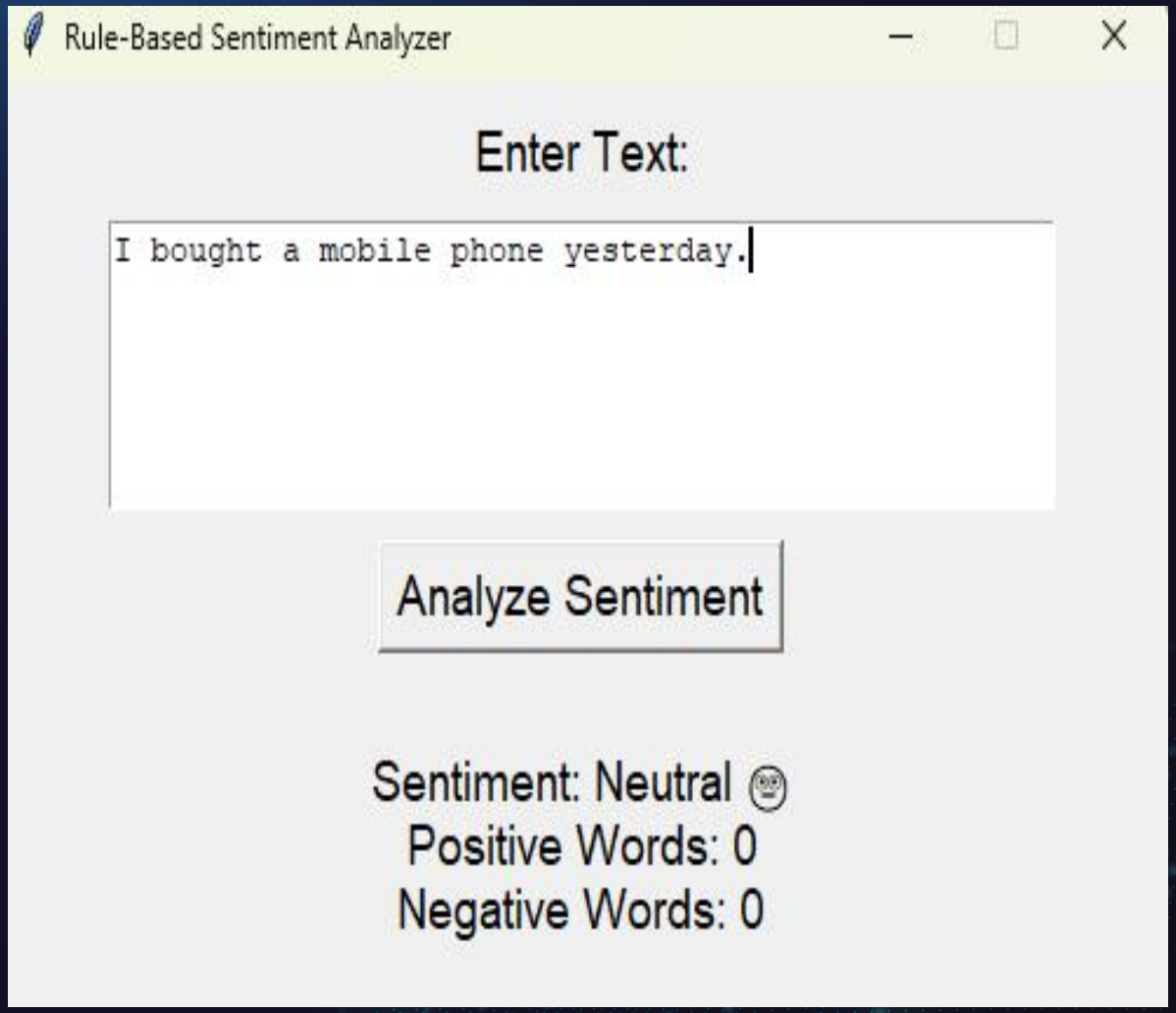
```python
# ---------- GUI ----------
root = Tk()
root.title("Rule-Based Sentiment Analyzer")
root.geometry("500x400")
root.resizable(False, False)

Label(root, text="Enter Text:", font=("Arial", 14)).pack(pady=10)

text_input = Text(root, height=6, width=50)
text_input.pack()

Button(
    root,
    text="Analyze Sentiment",
    font=("Arial", 14),
    command=analyze_sentiment
).pack(pady=10)

result_label = Label(
    root,
    text="Sentiment:",
```

```
63  result_label = Label(
64      root,
65      text="Sentiment:",
66      font=("Arial", 14)
67  )
68  result_label.pack(pady=20)
69
70  root.mainloop()
```

OUTPUT


Rule-Based Sentiment Analyzer
Enter Text:
This Project is amazing and nice
Analyze Sentiment
Sentiment: Positive
Positive Words: 2
Negative Words: 0


Rule-Based Sentiment Analyzer
Enter Text:
The experience was terrible and disappointing
Analyze Sentiment
Sentiment: Negative
Positive Words: 0
Negative Words: 1


Rule-Based Sentiment Analyzer
Enter Text:
I bought a mobile phone yesterday.
Analyze Sentiment
Sentiment: Neutral
Positive Words: 0
Negative Words: 0

PROJECT GOALS

Understand basic AI logic

Build GUI using Python

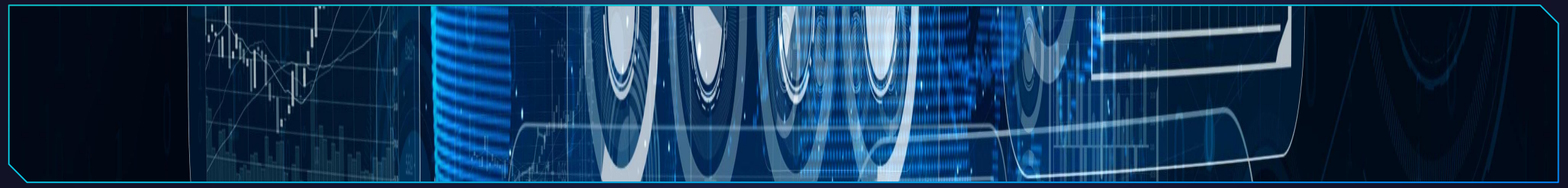
Implement rule-based reasoning

Analyze text sentiment automatically

Avoid heavy external libraries

Beginner-friendly AI project

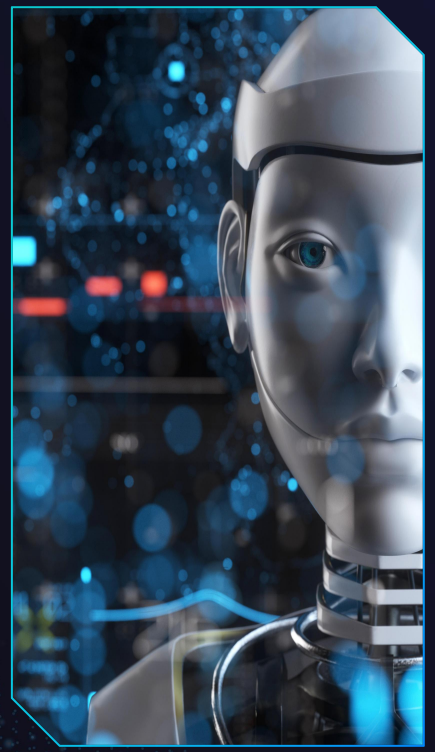
LIMITATIONS



- Works only on predefined positive and negative words
- Cannot understand new or unseen words
- Does not understand sentence context
- Fails to detect sarcasm or mixed emotions
- Accuracy depends on word list size
- Not suitable for complex language analysis

CONCLUSION

S



- Project successfully completed
- AI logic clearly implemented
- GUI makes system user-friendly
- Rule-based approach used
- Suitable for beginners
- Good foundation for AI learning

THANKS