AI MINI PROJECT

PRESENTED BY:

ABHIRUPMANDAL(RA2011003011105)

SHANTANU TRIPATHI((RA2011003011085)

ASHUTOSH SHARMA(RA2011003011084)

Title: Depression Detector- Sentiment Analysis and Mental State Prediction Project Report

Introduction

The purpose of this project is to design and implement a program that analyzes user input and predicts their mental state based on the input. The depression detector leverages sentiment analysis to evaluate users' thoughts and provides an assessment of their emotional well-being. By understanding the emotions conveyed through text, the application aims to contribute to mental health awareness and support.

Background

Mental health is a critical aspect of overall well-being, and detecting early signs of depression or other mental health issues is vital. Traditional methods, such as interviews and self-report questionnaires, have limitations and may not accurately capture a person's emotional state. Sentiment analysis offers an alternative approach by automatically evaluating the emotional tone of textual content, allowing for more frequent and less intrusive assessments.

Methodology

The program employs various libraries and techniques to analyze user input and predict mental states:

Preprocessing: Input text is preprocessed using regular expressions to eliminate mentions, URLs, special characters, and convert the text to lowercase for consistency.

Data loading: The labeled dataset of tweets is loaded using 'csv' and 'chardet' libraries, encoding the dataset and separating it into polarity (sentiment) and text components.

TextBlob: The 'TextBlob' library is utilized to classify the subjectivity of the input text, providing an additional dimension to the analysis.

Scikit-learn: A machine learning pipeline is constructed using 'CountVectorizer' for feature extraction and 'LogisticRegression' for classification. The sentiment analysis model is trained on the preprocessed dataset, and the trained model is saved using the 'pickle' library for future use.

Tkinter: A graphical user interface (GUI) is built with 'tkinter' library, allowing users to input their thoughts and receive feedback on their mental state.

Implementation

The program consists of several functions:

preprocess_text(): Preprocesses the input text, preparing it for analysis.

load_data(): Loads the dataset, returning the data as a list of dictionaries containing text and sentiment.

classify_subjectivity(): Determines the subjectivity of the text using TextBlob, providing an assessment of how subjective or objective the input is.

predict_sentiment_subjectivity(): Predicts sentiment and subjectivity for the given text, offering an analysis of the user's mental state.

submit_text(): Retrieves user input, predicts sentiment and subjectivity, and displays the results in the GUI.

The main section of the program creates the Tkinter GUI with a text entry field, a submit button, and a result label. The 'submit_text()' function is invoked when the user clicks the 'Analyze' button, presenting the sentiment analysis results.

CODE:

```
import os
import csv
import re
import chardet
import pickle
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from textblob import TextBlob
import tkinter as tk
from tkinter import ttk

def preprocess_text(text):
    text = re.sub(r'@[A-Za-z0-9]+', '', text)  # Remove mentions
```

```
    text = re.sub(r'https?://[A-Za-z0-9./]+', '', text)  # Remove URLs
    text = re.sub(r'[^A-Za-z0-9 ]+', '', text)  # Remove special characters
    text = text.lower()  # Convert text to lowercase
    return text

def load_data(filename):
    with open(filename, 'rb') as f:
        result = chardet.detect(f.read())
    encoding = result['encoding']
    data = []
    with open(filename, encoding=encoding) as csvfile:
        reader = csv.reader(csvfile)
        for row in reader:
            polarity = int(row[0])
            if polarity == 0:
                sentiment = 'negative'
            elif polarity == 2:
                sentiment = 'neutral'
            else:
                sentiment = 'positive'
            text = preprocess_text(row[5])
            data.append({'text': text, 'sentiment': sentiment})
    return data

def classify_subjectivity(text):
    blob = TextBlob(text)
    if blob.sentiment.subjectivity > 0.7:
        return 'very subjective'
    elif blob.sentiment.subjectivity > 0.5:
        return 'somewhat subjective'
    else:
        return 'objective'

# Create the pipeline
text_clf = Pipeline([
    ('vect', CountVectorizer(ngram_range=(1, 2))),
    ('clf', LogisticRegression(solver='liblinear', random_state=42)),
])

# Define the model file name
model_file = 'text_clf.pkl'

if not os.path.exists(model_file):
    # Train the model if it hasn't been trained yet
    train_data =
load_data(r'C:\Users\alexj\Downloads\trainingandtestdata\training.1600000.proc
essed.noemoticon.csv')
    X_train = [d['text'] for d in train_data]
```

```
    y_train = [d['sentiment'] for d in train_data]
    text_clf.fit(X_train, y_train)

    # Save the trained model
    with open(model_file, 'wb') as f:
        pickle.dump(text_clf, f)

# Load the saved model
with open(model_file, 'rb') as f:
    text_clf = pickle.load(f)

def predict_sentiment_subjectivity(text):
    sentiment = text_clf.predict([text])[0]
    subjectivity = classify_subjectivity(text)
    return sentiment, subjectivity

def submit_text():
    user_input = text_entry.get()
    predicted_sentiment, predicted_subjectivity =
predict_sentiment_subjectivity(user_input)
    result.set(f"The sentiment analysis indicates that you are feeling
{predicted_sentiment} and your thoughts are {predicted_subjectivity}.")

# Create the main window
root = tk.Tk()
root.title("Sentiment Analysis")

# Create a label for the input field
text_label = ttk.Label(root, text="Please enter your thoughts:")
text_label.pack(padx=10, pady=10)

text_entry = ttk.Entry(root, width=50)
text_entry.pack(padx=10, pady=10)

# Create a submit button
submit_button = ttk.Button(root, text="Analyze", command=submit_text)
submit_button.pack(padx=10, pady=10)

# Create a label to display the result
result = tk.StringVar()
result_label = ttk.Label(root, textvariable=result)
result_label.pack(padx=10, pady=10)

# Run the main loop to display the window
root.mainloop()
```

Evaluation

The effectiveness of the depression detector was evaluated by comparing its predictions with actual sentiment labels in the dataset. The model achieved high accuracy, indicating its ability to reliably classify the sentiment and subjectivity of the input text. However, it is essential to consider that the dataset used is composed of tweets and may not be representative of all types of textual content that users may input. Further evaluation using more diverse data sources is recommended to ensure robustness and applicability.

Limitations and Challenges

While the depression detector shows promise in sentiment analysis and mental state prediction, there are some limitations and challenges to consider:

Data Representation: The current dataset consists of tweets, which may not represent the full range of textual content that users input. Expanding the dataset to include more diverse sources, such as blog posts, forum discussions, and personal journal entries, could improve the model's performance.

Context Sensitivity: Sentiment analysis may not always capture the nuances of human emotions, particularly in cases where sarcasm, irony, or cultural context plays a role. Addressing these challenges may require more advanced natural language processing techniques or incorporating additional contextual information.

Language Support: The current implementation only supports English text. Expanding the program to handle other languages would make it more inclusive and accessible to a broader audience.

Privacy Concerns: Users may be hesitant to share personal thoughts and emotions with an automated system, particularly if there are concerns about data storage or potential misuse. Ensuring privacy and data protection is crucial to build trust and promote widespread adoption.

Results

The depression detector offers a user-friendly interface that enables users to enter their thoughts and receive an assessment of their emotional state. The program accurately classifies sentiment and subjectivity, providing valuable insights into users' mental well-being.

Conclusion and Future Work

This project demonstrates the successful development of a depression detector using sentiment analysis techniques. The program effectively evaluates user input, classifying sentiment and subjectivity, and offers feedback on the user's emotional state. The application could be extended to other mental health-related scenarios, integrated into various platforms,

or adapted to monitor social media content to provide early warning signs for individuals experiencing mental health challenges.