In [ ]:
```
import pandas as pd
import numpy as np
import nltk
from nltk.corpus import stopwords
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.utils import to_categorical
from sklearn.metrics import accuracy_score, classification_report

nltk.download('stopwords')
stop_words = set(stopwords.words('english'))

def load_dataset(file_path):
    try:
        df = pd.read_csv(file_path, encoding='latin1')
        return df
    except FileNotFoundError:
        print(f"Error: File '{file_path}' not found.")
        raise

def preprocess_data(df):
    df['message to examine'] = df['message to examine'].fillna('')
    df['label (depression result)'] = df['label (depression result)'].fillna('neutr
    df['label (depression result)'] = df['label (depression result)'].astype(str)
    df['message to examine'] = df['message to examine'].apply(lambda x: " ".join([w
    return df

def train_model(X_train, y_train):
    vectorizer = TfidfVectorizer(max_features=5000)
    X_train_vec = vectorizer.fit_transform(X_train).toarray()

    label_encoder = LabelEncoder()
    y_train_encoded = label_encoder.fit_transform(y_train)
    y_train_one_hot = to_categorical(y_train_encoded)

    model = Sequential()
    model.add(Dense(512, input_dim=X_train_vec.shape[1], activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(256, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(y_train_one_hot.shape[1], activation='softmax'))

    model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accu
    model.fit(X_train_vec, y_train_one_hot, epochs=10, batch_size=32, validation_sp

    return model, vectorizer, label_encoder

def evaluate_model(model, vectorizer, label_encoder, X_test, y_test):
    X_test_vec = vectorizer.transform(X_test).toarray()

    y_test_encoded = label_encoder.transform(y_test)
    y_test_one_hot = to_categorical(y_test_encoded)

    y_pred_prob = model.predict(X_test_vec)
    y_pred = np.argmax(y_pred_prob, axis=1)

    accuracy = accuracy_score(np.argmax(y_test_one_hot, axis=1), y_pred)
    report = classification_report(np.argmax(y_test_one_hot, axis=1), y_pred, targe
```

```
    return accuracy, report

def predict_sentiment(model, vectorizer, label_encoder, text):
    text = " ".join([word for word in text.split() if word not in stop_words])
    text_vec = vectorizer.transform([text]).toarray()
    text_pred_prob = model.predict(text_vec)
    text_pred = np.argmax(text_pred_prob, axis=1)
    sentiment = label_encoder.inverse_transform(text_pred)
    return sentiment[0]

def main(file_path):
    df = load_dataset(file_path)
    df = preprocess_data(df)

    X = df['message to examine']
    y = df['label (depression result)']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random

    print("Training data shape:", X_train.shape)
    print("Testing data shape:", X_test.shape)

    model, vectorizer, label_encoder = train_model(X_train, y_train)
    accuracy, report = evaluate_model(model, vectorizer, label_encoder, X_test, y_t

    print(f"Accuracy: {accuracy}")
    print(f"Classification Report:\n{report}")

    print("\nLabel encoding mapping:")
    for index, label in enumerate(label_encoder.classes_):
        print(f"{index}: {label}")

    while True:
        user_input = input("Enter a message to analyze sentiment (or type 'exit' to
        if user_input.lower() == 'exit':
            break
        sentiment = predict_sentiment(model, vectorizer, label_encoder, user_input)
        if sentiment == 'positive':
            print(f"The sentiment of the message is: Positive")
        elif sentiment == 'negative':
            print(f"The sentiment of the message is: Negative")
        else:
            print(f"The sentiment of the message is: Neutral")


main('sentiment_tweets3.csv')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\jalum\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
C:\Users\jalum\anaconda3\Lib\site-packages\keras\src\layers\core\dense.py:87: User
Warning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using
Sequential models, prefer using an `Input(shape)` object as the first layer in the
model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

```
Training data shape: (8251,)
Testing data shape: (2063,)
Epoch 1/10
233/233 ━━━━━━━━━━━━━━━━━━━━ 7s 25ms/step - accuracy: 0.8248 - loss: 0.3760 - val_
accuracy: 0.9855 - val_loss: 0.0323
Epoch 2/10
233/233 ━━━━━━━━━━━━━━━━━━━━ 5s 23ms/step - accuracy: 0.9948 - loss: 0.0182 - val_
accuracy: 0.9927 - val_loss: 0.0253
Epoch 3/10
233/233 ━━━━━━━━━━━━━━━━━━━━ 5s 23ms/step - accuracy: 0.9990 - loss: 0.0040 - val_
accuracy: 0.9939 - val_loss: 0.0236
Epoch 4/10
233/233 ━━━━━━━━━━━━━━━━━━━━ 6s 24ms/step - accuracy: 0.9997 - loss: 0.0021 - val_
accuracy: 0.9891 - val_loss: 0.0255
Epoch 5/10
233/233 ━━━━━━━━━━━━━━━━━━━━ 6s 25ms/step - accuracy: 0.9999 - loss: 8.0786e-04 -
val_accuracy: 0.9879 - val_loss: 0.0331
Epoch 6/10
233/233 ━━━━━━━━━━━━━━━━━━━━ 6s 27ms/step - accuracy: 0.9994 - loss: 0.0034 - val_
accuracy: 0.9855 - val_loss: 0.0380
Epoch 7/10
233/233 ━━━━━━━━━━━━━━━━━━━━ 6s 26ms/step - accuracy: 1.0000 - loss: 4.5293e-04 -
val_accuracy: 0.9879 - val_loss: 0.0447
Epoch 8/10
233/233 ━━━━━━━━━━━━━━━━━━━━ 6s 26ms/step - accuracy: 0.9996 - loss: 0.0023 - val_
accuracy: 0.9867 - val_loss: 0.0441
Epoch 9/10
233/233 ━━━━━━━━━━━━━━━━━━━━ 6s 25ms/step - accuracy: 1.0000 - loss: 3.9733e-04 -
val_accuracy: 0.9818 - val_loss: 0.0616
Epoch 10/10
233/233 ━━━━━━━━━━━━━━━━━━━━ 6s 24ms/step - accuracy: 0.9999 - loss: 0.0012 - val_
accuracy: 0.9831 - val_loss: 0.0529
65/65 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
Accuracy: 0.983034415899176
Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      1614
           1       0.97      0.95      0.96       449

    accuracy                           0.98      2063
   macro avg       0.98      0.97      0.97      2063
weighted avg       0.98      0.98      0.98      2063


Label encoding mapping:
0: 0
1: 1
Enter a message to analyze sentiment (or type 'exit' to quit): I am just feeling o
kay today.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 15ms/step
The sentiment of the message is: Neutral
```

In [ ]: