MINI Project Report 2022-2023 3rd Semester

PROBLEM STATEMENT:

Sentimental analysis of tweets.

Team Members:

Amruta-1CR21AD010

Malavika-1CR21AD031

Shanvi-1CR21AD050

Index

Abstract ,

The goal of tweet sentiment analysis is to find the positive, negative, or neutral sentiment part in the twitter data. Sentiment analysis can help any organization to find people's opinions of their company and products. We have applied sentiment analysis on twitter data set. Our model takes input tweet, sentiment, and output selected text starting and ending in input tweet. We are using the tensorflow with LSTM and keras in python.

With the recent advances in deep learning, the ability of algorithms to analyse the text has improved a lot. Creative use of advanced artificial intelligence techniques can be an effective tool for doing in-depth research.

Learning Objectives, Tasks Handled

This project gives us a knowledge about machine learning. Machine learning chains together algorithms that aim to simulate how the human brain works, otherwise known as an artificial neural network, and has enabled many practical applications of machine learning, including customer support automation and self-driving cars.it also helps to improve our coding skills. Understanding the machine learning algorithms are very useful to build any ai model .this will be very helpful in future.

Tasks handled:

- Creating a dataset for training and test using python.
- Preparing the dataset for training using keras(tensorflow).
- Creating a model of dataset.
- Evaluation of the model to predict the sentiment.

Implementation:

Code:

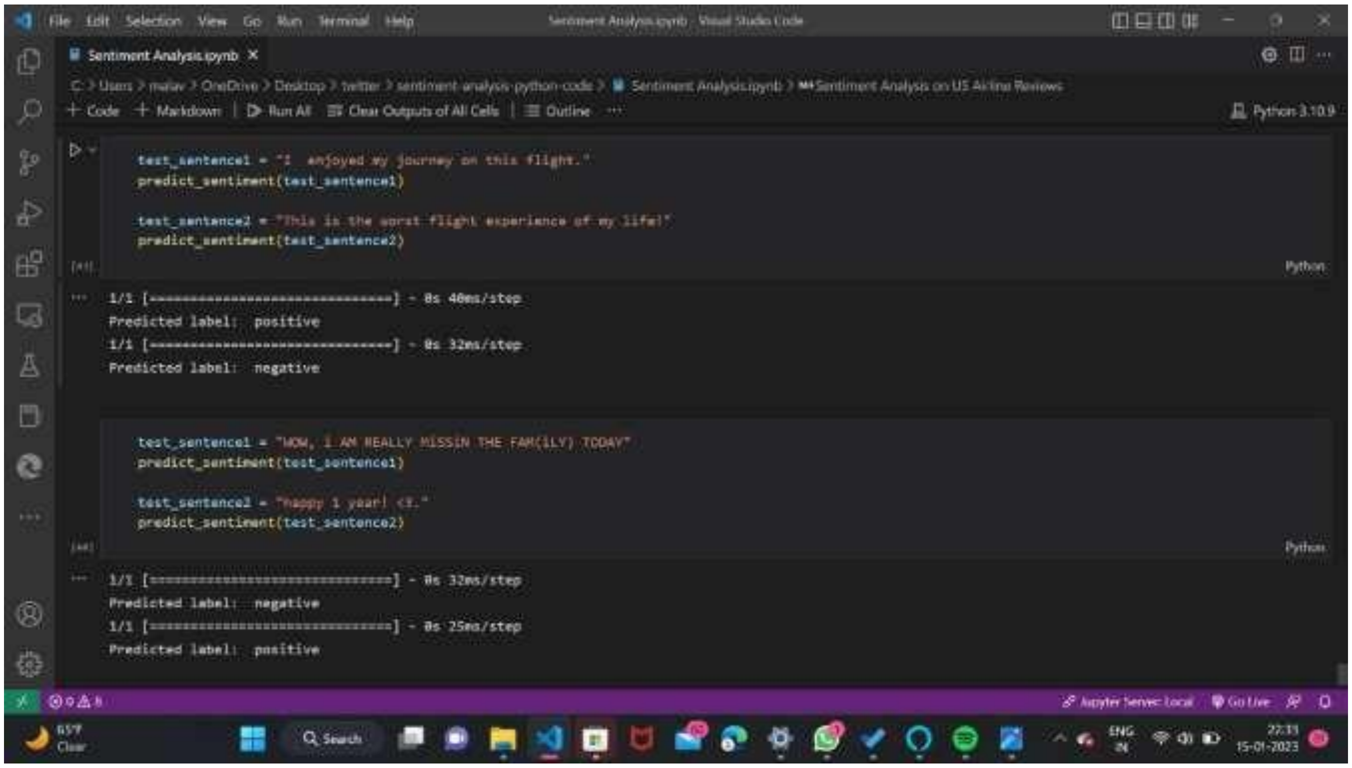
```
#importing the python library functions.
import pandas as pd
import matplotlib.pyplot as plt

from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM,Dense, Dropout, SpatialDropout1D
from tensorflow.keras.layers import Embedding



import os

os.chdir('C:\\Users\\malav\\OneDrive\\Desktop')

os.getcwd()

#reading the dataset

df = pd.read_csv("Tweets.csv")

df.head()

df.columns

tweet_df = df[['text','sentiment','textID']]
print(tweet_df.shape)
tweet_df.head(5)
#eliminating a columns with neutral sentiments.
tweet_df = tweet_df[tweet_df['sentiment'] != 'neutral']
```

```
print(tweet_df.shape)
tweet_df.head(5)
#counting the number of positive and negative sentiments in data



tweet_df["sentiment"].value_counts()

tweet = tweet_df.text.value
```

```
#preparing the dataset for training
tokenizer = Tokenizer(num_words=5000)tokenizer.fit_on_texts(tweet)
    vocab_size = len(tokenizer.word_index) + 1
    encoded_docs = tokenizer.texts_to_sequences(tweet)
    padded_sequence = pad_sequences(encoded_docs, maxlen=200)


    print(tweet[0])
    print(encoded_docs[0])


    print(padded_sequence[0])

#creating a model of dataset.
    embedding_vector_length = 32
    model = Sequential()
    model.add(Embedding(vocab_size, embedding_vector_length, input_length=200) )
    model.add(SpatialDropout1D(0.25))
    model.add(LSTM(50, dropout=0.5, recurrent_dropout=0.5))
    model.add(Dropout(0.2))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(loss='binary_crossentropy',optimizer='adam', metrics=['accuracy'])
    print(model.summary())
#applying loss and optimization to a model.
    history   =   model.fit(padded_sequence,sentiment_label[0],validation_split=0.2,
    epochs=5, batch_size=32)
#plotting a graph

    plt.plot(history.history['accuracy'], label='acc')
    plt.plot(history.history['val_accuracy'], label='val_acc')
    plt.legend()
    plt.show()
    plt.savefig("Accuracy plot.jpg")
    plt.plot(history.history['loss'], label='loss')
    plt.plot(history.history['val_loss'], label='val_loss')
    plt.legend()
    plt.show()
    plt.savefig("Loss plot.jpg")
#saving the model and prediction of data

    def predict_sentiment(text):
        tw = tokenizer.texts_to_sequences([text])
        tw = pad_sequences(tw,maxlen=200)
        prediction = int(model.predict(tw).round().item())
        print("Predicted label: ", sentiment_label[1][prediction])

    test_sentence1 = "WOW, i AM REALLY MiSSiN THE FAM(iLY) TODAY"
    predict_sentiment(test_sentence1)

    test_sentence2 = "happy 1 year! <3."
    predict_sentiment(test_sentence2)



    neg = df[df['sentiment']=='negative']
    pos = df[df['sentiment']=='positive']


    import plotly.graph_objs as go
    fig = go.Figure()
    for col in pos.columns:
        fig.add_trace(go.Scatter(x=pos['sentiment'], y=pos['textID'],
                            name = col,
                            mode = 'markers+lines',
                            line=dict(shape='linear'),
                            connectgaps=True,
                            line_color='green'
                            )
                )
```

```
for col in neg.columns:
   fig.add_trace(go.Scatter(x=neg['sentiment'], y=neg['textID'],
                        name = col,
                        mode = 'markers+lines',
                        line=dict(shape='linear'),
                        connectgaps=True,
                        line_color='red'
                        )

              )
fig.show()
```

Output:



```
test_sentence1 = "I  enjoyed my journey on this flight."
predict_sentiment(test_sentence1)

test_sentence2 = "This is the worst flight experience of my life!"
predict_sentiment(test_sentence2)
```

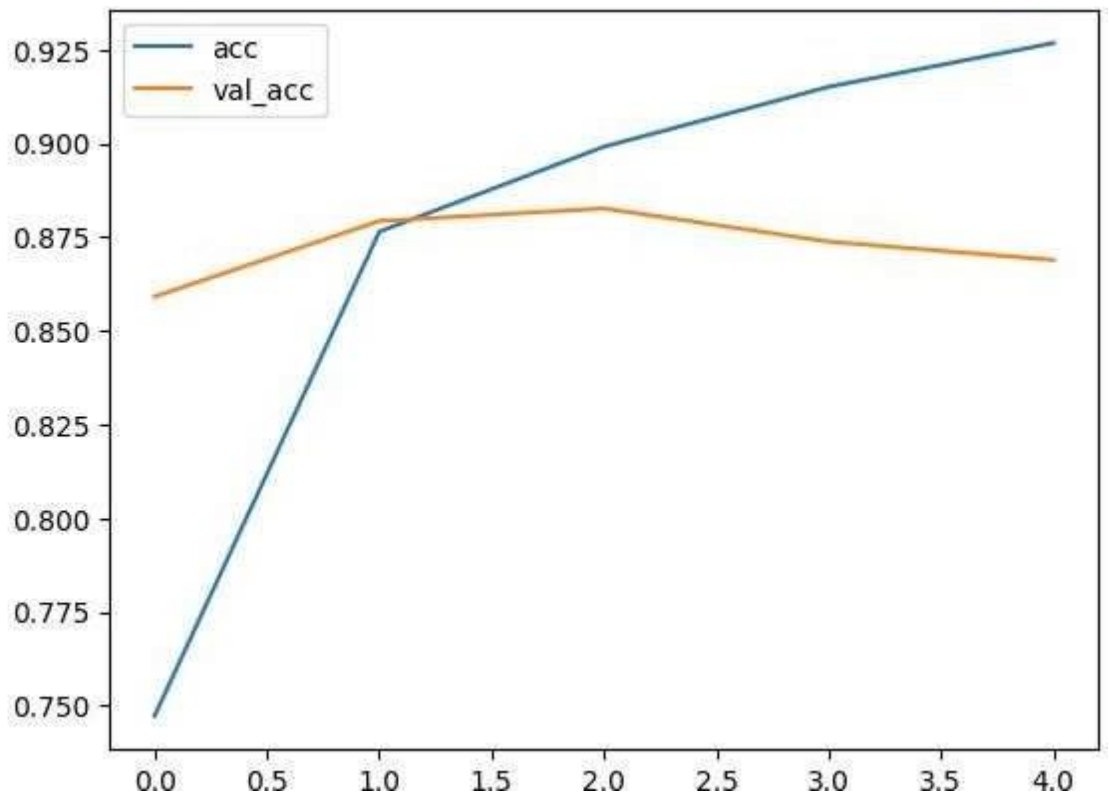
```
1/1 [==============================] - 0s 40ms/step
Predicted label:  positive
1/1 [==============================] - 0s 32ms/step
Predicted label:  negative
```

```
test_sentence1 = "WOW, I AM REALLY MISSIN THE FAM(ILY) TODAY"
predict_sentiment(test_sentence1)

test_sentence2 = "happy 1 year! <3."
predict_sentiment(test_sentence2)
```

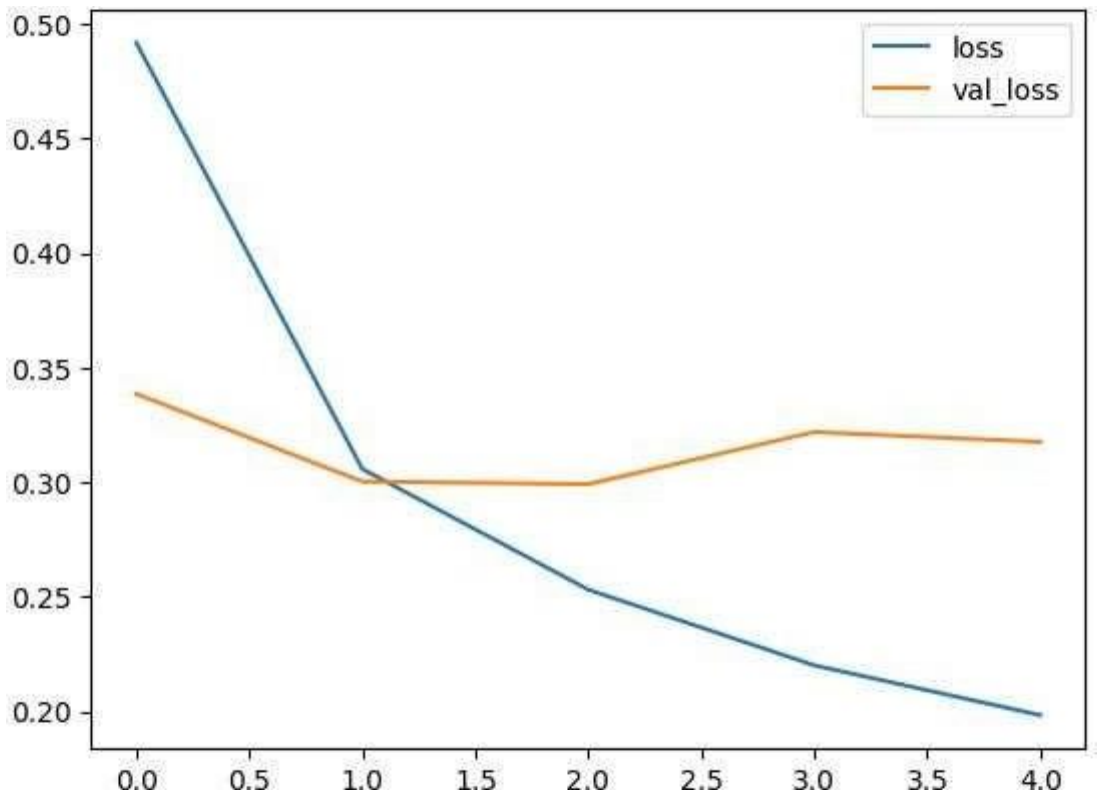
```
1/1 [==============================] - 0s 32ms/step
Predicted label:  negative
1/1 [==============================] - 0s 25ms/step
Predicted label:  positive
```

Prediction of the sentiment of the data by entering the text.

Accuracy graph: increasing in the value of accuracy or accuracy percentage as we go through the model.(x-epoch,y-accuracy)



Loss graph: decreasing in the loss percentage as we move through the model .(x-epoch, y-loss)



Conclusion:

Sentiment analysis offers undeniable analytical results, whether from regular documents, business reports, social media monitoring, customer support tickets, and more. And deep learning allows you to put more powerful algorithms and more tools to work on your data. When employed with user-friendly and in-depth visualization tools, you can create captivating data stories to prove your brand's worth and help push your business forward.. and take the world forward with you.