Submission: 5/5/2024

NLU PROJECT

Sentiment Analysis

Team Members
Ahmed Gamal Ahmed_20221459969
Myar Hany Mohamed_20221376556
Mona Badr Mostafa_2103111

TABLE OF CONTENTS

DATASET

Sentiment Analysis idea is to recognize the sentiment (emotion) of an input sentence.

We have used a Dataset with 20000 Row. A row consists of Sentence and Label. This Dataset is called Twitter Emotion Classification.

Each Tweet (Sentence) has a label from these classes:
Joy, Anger, Sadness, Fear, Love, Surprise
But Love and Surprise classes had few sentences so we dropped these 2 classes and used the other 4 classes.

Here is the Dataset Insights:

LABEL	# OF SENTENCES
JOY	6761
SADNESS	5797
ANGER	2709
FEAR	2373

Link of Dataset on Kaggle (Twitter Emotion Classification):
https://www.kaggle.com/code/shtrausslearning/twitter-emotion-classification/input?select=training.csv

TEXT PREPROCESSING

We have created a function called preprocess text that converts all of the sentences inside our dataset to a preprocessed version that will be used in training and testing

No. 01 – Lowercasing

Returns a lower cased version from the text
Ex: I Am The Happiest --> i am the happiest

No. 02 – Tokenization

Splitting the sentence into tokens (words)
Ex: i am the happiest to {"i", "am", "the","happiest"}

No. 03 – Stop Words

Removing all stop words from the sentence
Ex: {"i", "am", "the","happiest"} to {"i",happiest"}

No. 04 – Lemmatization

Returning the words to their lemma (the form in dictionary)
Ex: {"i",happiest"} to {"i",happy"}

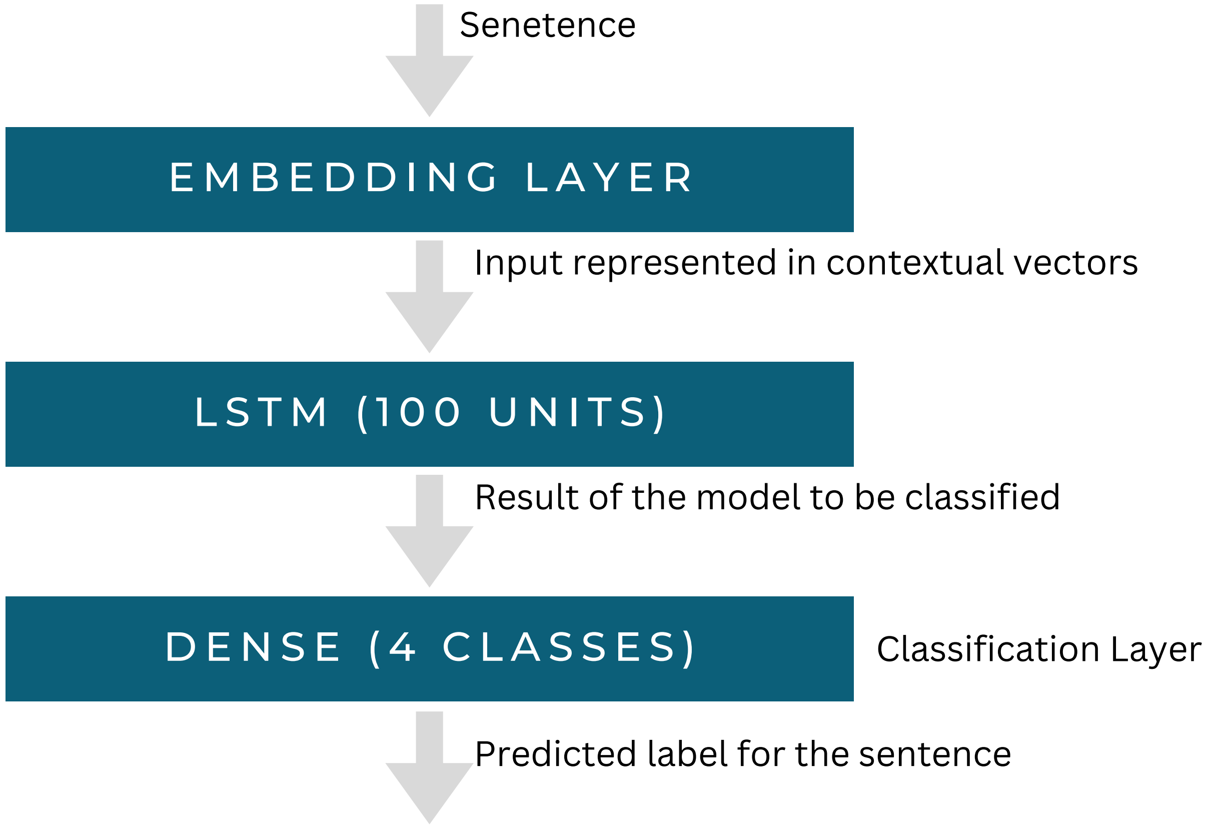
LSTM MODEL

We have splitted that data for:

- 80% Training Set
- 20% Testing Set

We used **Padding** on the sentences to make all of them in the same length.
We also used **OOV Tokenizer** which tokenize the words of the sentence and give the <oov> tag to any unseen words in the training.

MODEL STRUCTURE



Senetence

EMBEDDING LAYER

Input represented in contextual vectors

LSTM (100 UNITS)

Result of the model to be classified

DENSE (4 CLASSES)

Classification Layer

Predicted label for the sentence

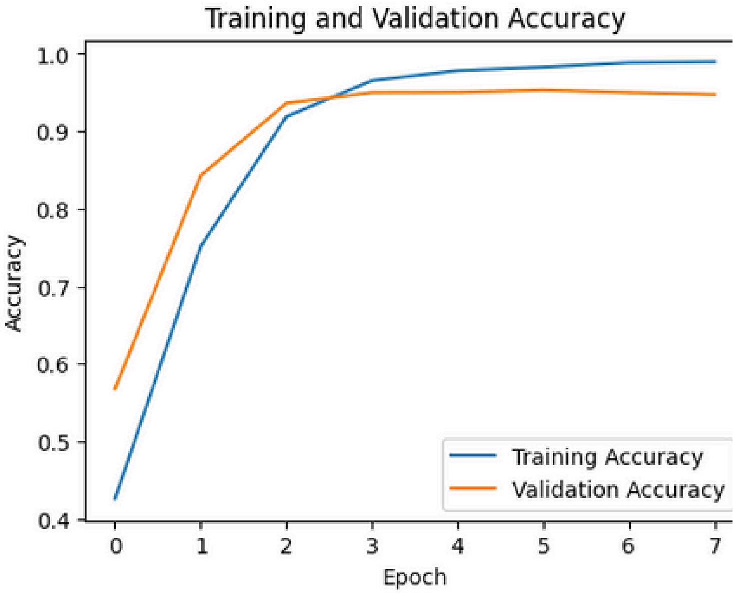
MODEL EVALUATION

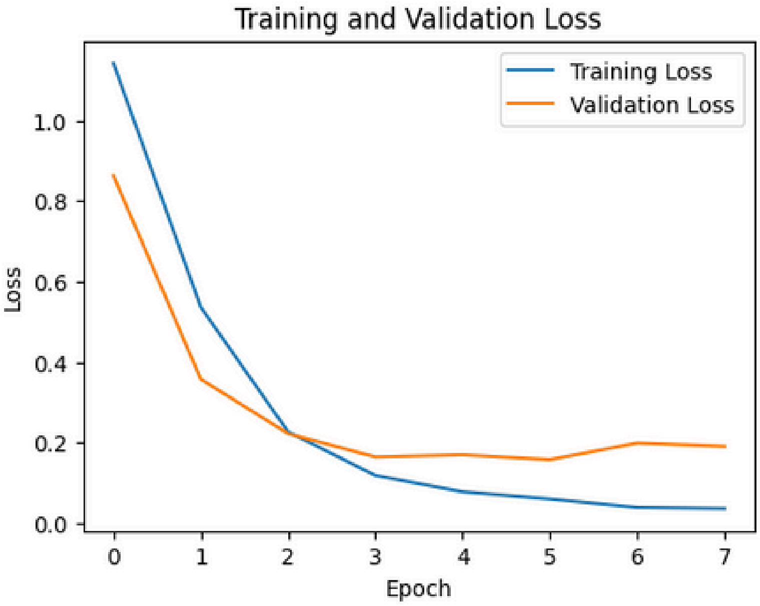
Our model trained on 8 Epochs after Early Stopping with these losses in training and validation:

- Training Loss: 0.0603
- Validation Loss: 0.1577

And got these training and validation accuracies:

- Training Accuracy: 98.26%
- Validation Accuracy: 95.29%





NAIIVE BAYES MODEL

We have splitted that data for:

- 80% Training Set
- 20% Testing Set

We used **TF-IDF Vectorizer** on the sentences to create features matrix on both uni-grams and bi-grams.
We also used **Multi Nomial Naiive Bayes** with **alpha = 0.5.**

This model has got these accuracies:

- Training Accuracy: 99.84%
- Validation Accuracy: 85.29%

Likelihood

Class Prior Probability

$$P(c \mid x) = \frac{P(x \mid c)\,P(c)}{P(x)}$$

Posterior Probability

Predictor Prior Probability

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

LOGISTIC REGRESSION

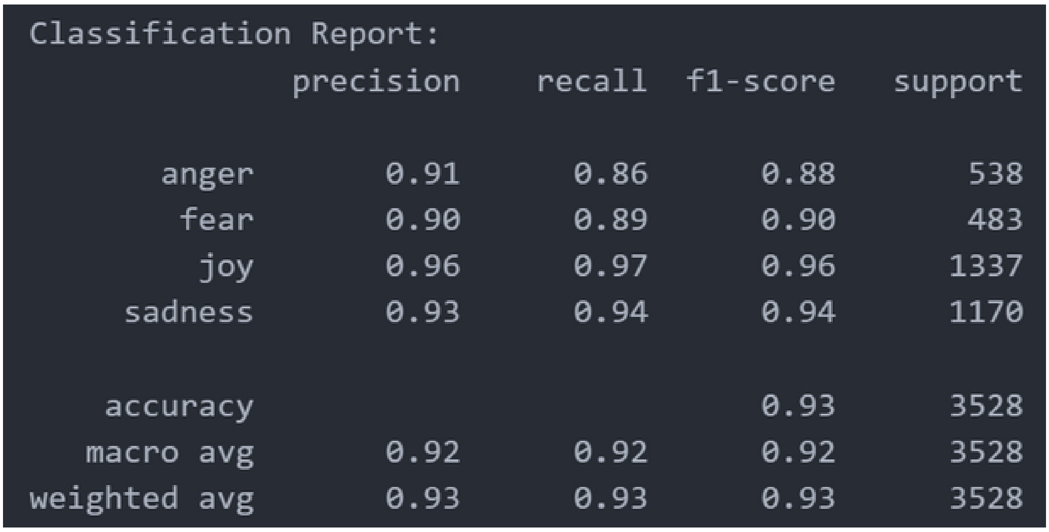
We have splitted that data for:

- 80% Training Set
- 20% Testing Set

We used **Count Vectorizer** on the sentences to create sparse matrix representations.
The Logistic Regression is trained using **Maximum Number of Iterations = 20000 Iteration**

This model has got these accuracies:

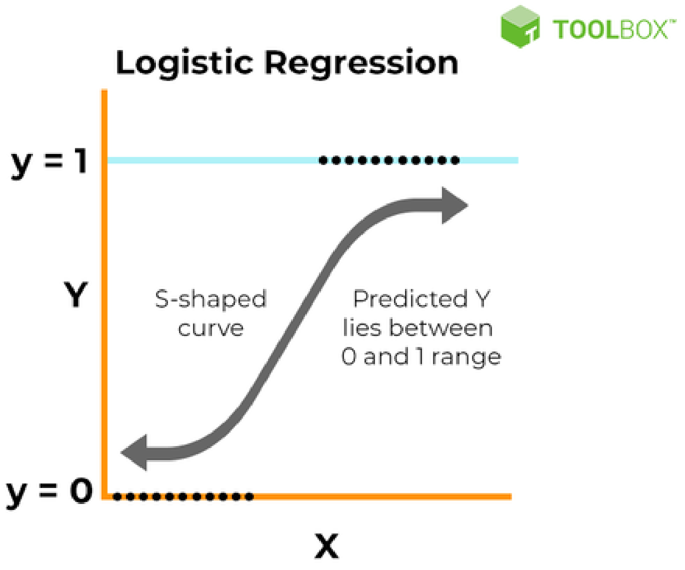
- Validation Accuracy: 93.45%



```
Classification Report:
              precision    recall  f1-score   support

       anger       0.91      0.86      0.88       538
        fear       0.90      0.89      0.90       483
         joy       0.96      0.97      0.96      1337
     sadness       0.93      0.94      0.94      1170

    accuracy                           0.93      3528
   macro avg       0.92      0.92      0.92      3528
weighted avg       0.93      0.93      0.93      3528
```



MODELS COMPARISON

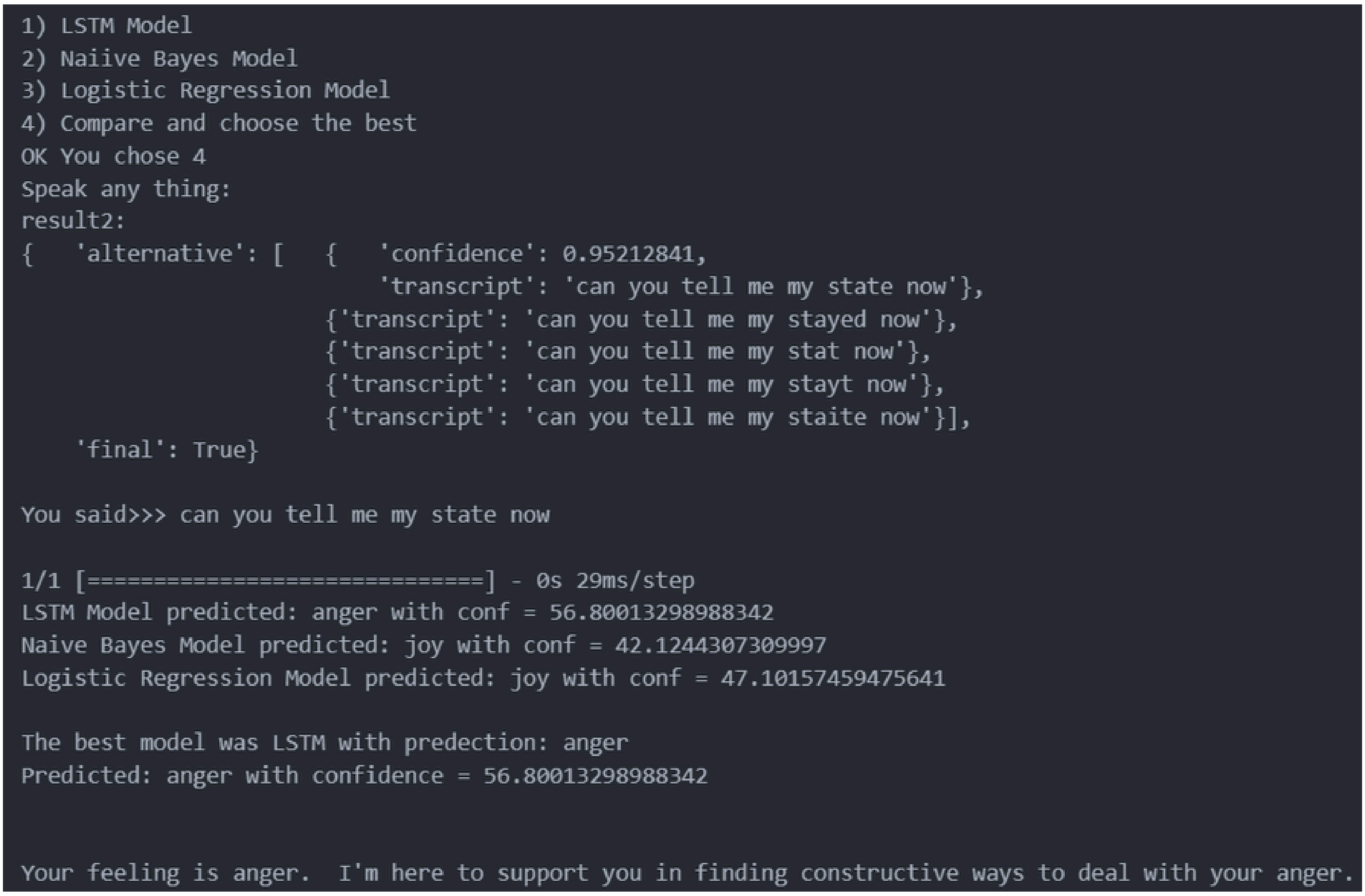
MODEL	TESTING ACCURACY
LSTM	95.29%
NAIIVE BAYES	85.29%
LOGISTIC REGRESSION	93.45%

So in conclusion, the **LSTM Model** has the maximum accuracy on the validation set, and next to it is the **Logitistic Regression Model** and last is the **Naiive Bayes Model**

SPEECH RECOGNITION

We used Speech Recognition that takes the input from the speaker from speech to text that will be the input for the models.

We also used pytts to do the task of Text To Speech for interactive sounds (as if it’s a bot) and also respond to our feeling with a quote or another sentence.



```
1) LSTM Model
2) Naiive Bayes Model
3) Logistic Regression Model
4) Compare and choose the best
OK You chose 4
Speak any thing:
result2:
{   'alternative': [   {   'confidence': 0.95212841,
                           'transcript': 'can you tell me my state now'},
                       {'transcript': 'can you tell me my stayed now'},
                       {'transcript': 'can you tell me my stat now'},
                       {'transcript': 'can you tell me my stayt now'},
                       {'transcript': 'can you tell me my staite now'}],
    'final': True}

You said>>> can you tell me my state now

1/1 [==============================] - 0s 29ms/step
LSTM Model predicted: anger with conf = 56.80013298988342
Naive Bayes Model predicted: joy with conf = 42.1244307309997
Logistic Regression Model predicted: joy with conf = 47.10157459475641

The best model was LSTM with predection: anger
Predicted: anger with confidence = 56.80013298988342


Your feeling is anger.  I'm here to support you in finding constructive ways to deal with your anger.
```