University of Jeddah
Collage of computer science &
Artificial Intelligence

Analysis of Arabic 100k Reviews

Phase3

Final Submission

Team Members:	
Hadeel Almutairi	2117022
Renad Alzahrani	2110292
Rawan Jaafari	2111481

Instructor:
Latifah Aljiffry

Abstract:

Since text analysis in the Arabic language requires an understanding of the distinctive grammatical, syntactic, and semantic rules of the language, comprehending the grammatical structures, sentence patterns, and linguistic formulas used in Arabic, such as conjugation, particles, and pronoun structures, is essential for understanding and analyzing Arabic texts. Additionally, there are unique cultural and expressive challenges that need to be addressed when analyzing emotions and feelings expressed in Arabic. Cultural differences must be taken into account, as the concept of emotions and feelings may vary from one culture to another. Some cultural factors to consider include values, beliefs, customs, and traditions related to expressing emotions in Arabic. These efforts in analyzing Arabic texts aim to enable a deeper understanding of texts and extract encoded meanings and emotions, contributing to improving comprehension and interaction with these texts. Advanced techniques in natural language processing, such as deep learning and machine learning, can be used to enhance the performance of Arabic text analysis in terms of accuracy and interpretation.

Introduction:

Natural Language Processing (NLP) is a vital field of artificial intelligence that aims to understand and process human language in an interactive and intelligent manner. Analyzing sentiments in Arabic language reviews is one of the key challenges in the field of Neural Language Programming, as it involves unique linguistic and cultural complexities. In this project, we applied NLP techniques to a dataset of "100,000 Arabic reviews" to analyze sentiments in Arabic reviews. Our main goal in this project was to build a model capable of accurately analyzing sentiments in Arabic reviews. Firstly, we performed data preprocessing, which involved removing stop words, tokenization, and stemming. We utilized different models to achieve this goal, such as Long Short-Term Memory (LSTM) model in deep learning, and logistic regression and multinomial logistic regression models in machine learning. However, we encountered the challenge of accurate sentiment classification in Arabic reviews due to the linguistic and cultural complexities of the Arabic language. These complexities include variations in dialects, idiomatic expressions, and different cultural contexts that impact sentiment analysis. To address this issue, we evaluated the performance of the utilized models by comparing their accuracy. The results showed that the "logistic regression" model achieved better performance, while the "MultinomialNB" model performed less effectively. This indicates that the logistic regression model is more suitable for sentiment analysis in Arabic reviews. Additionally, we employed the Term Frequency-Inverse Document Frequency (TF-IDF) technique to transform the textual data into a TF-IDF matrix. This technique represents the importance of words in the reviews based on their frequency in the document and inverse frequency in other documents. This approach helps in identifying the significance of words in the reviews and improving the accuracy of sentiment analysis.

Background/Related Work:

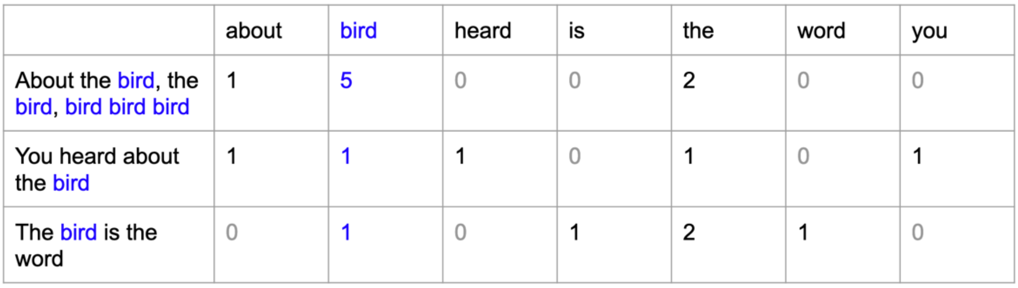
Natural language processing (NLP) is one of the fields that seeks to understand and process human language using computer programs. So, this means that computers can understand human language and apply some machine learning and deep learning techniques to it to learn it. This research [1] aims to analyze the emotions that can be found in tweets by using the logistic regression algorithm, which is one of the machine learning techniques that can be trained on a dataset to identify emotional text. As a result of this algorithm, better performance in sentiment analysis is achieved and high accuracy in emotion prediction can be achieved. Support vector machines (SVMs) are another algorithm used to identify patterns according to linear classification, which is another way to identify patterns. In addition, this research[2] aims to analyze emotions using a method called long-term memory (LSTM), which is a type of deep neural network that is commonly used to classify text using deep neural networks that are used to analyze emotions. As a result of their ability to capture context in texts as well as sequence in texts, LSTM models have become widely popular in sentiment analysis. This research [3] aims to analyse the sentiment of customer comments in telecommunications services so that the sentiment is classified as negative, positive, or neutral using the Multinomial Naïve Bayes classification model, which is considered one of the machine learning techniques, as it classifies the text based on the distribution of words. Moreover, the classification achieves the highest accuracy. It is considered one of the most effective models in classification

Approach:

Problem Statement: Sentiment analysis is the process of analyzing a written text and trying to understand it to classify the text into one of three categories, either positive text, negative text, or natural text meaning that is neither negative nor positive.

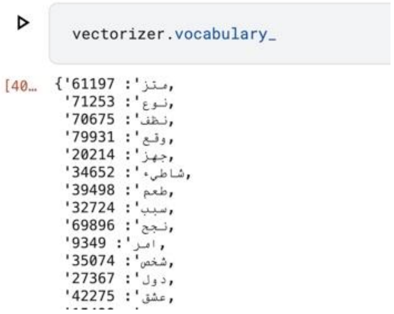
Feature Representation:

Bag of Words (BoW) is a natural language processing technique for extracting features from text. BoW models are an unstructured collection of all known words in a text document. The words are selected mainly based on frequency, ignoring word order and context. Table below shows the mechanism of how bag of words works.



	about	bird	heard	is	the	word	you
About the bird, the bird, bird bird bird	1	5	0	0	2	0	0
You heard about the bird	1	1	1	0	1	0	1
The bird is the word	0	1	0	1	2	1	0

Bag of words sample of our dataset:

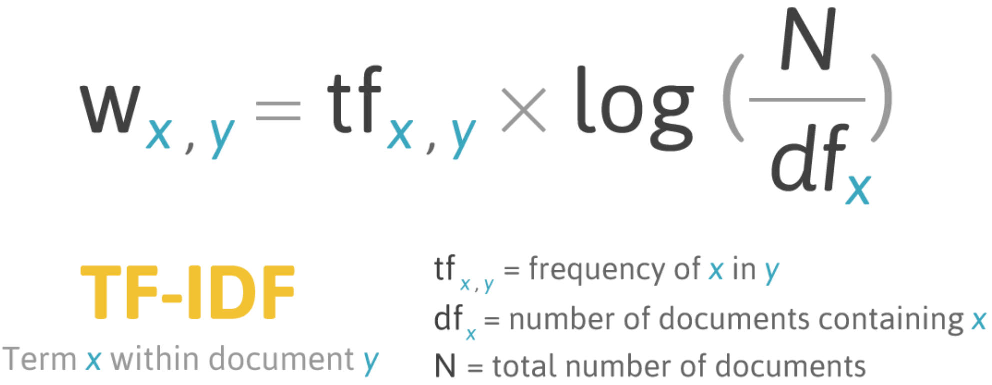


```
vectorizer.vocabulary_
```

```
[40… {'متز': 61197,
      'نوع': 71253,
      'نظف': 70675,
      'وقع': 79931,
      'جهز': 20214,
      'شاطىء': 34652,
      'طعم': 39498,
      'سبب': 32724,
      'نجح': 69896,
      'امر': 9349,
      'شخص': 35074,
      'دول': 27367,
      'عشق': 42275,
```

Term Frequency-Inverse Document Frequency

TF-IDF is an abbreviation Term Frequency-Inverse Document Frequency. It is a numerical statistic that is used to determine the importance of a word in a document relative to a collection of documents. TFIDF gives weights with higher values to terms that are rare in the text but common in the document.



$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

$tf_{x,y}$ = frequency of x in y

df_x = number of documents containing x

N = total number of documents

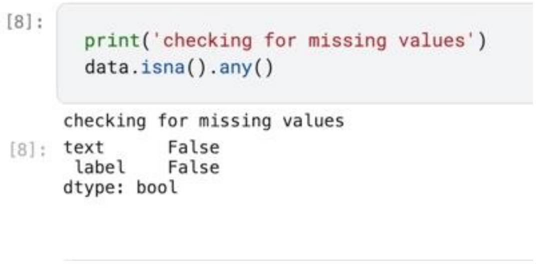
Word Embedding: it is a natural language processing technique; words will be represented as dense vectors of real numbers. By capturing the semantic relations between words, these vectors allow models to understand the context and meaning of words based on how they are used. Word embedding is based on the core idea that words with similar vector representations should also have similar words, which helps models generalize better to new or unseen data.

Text Vectorization: Text vectorization is the act of transforming textual data into numerical representations so that models can process and learn from natural language data efficiently. TensorFlow has a function called text vectorization that makes it easier to transform a collection of strings into token sequences or dense representations. Preprocessing operations like padding, lowercasing, and tokenization are all part of this process. In NLP models, text vectorization functions as a preprocessing layer, converting unprocessed textual data into a format that can be fed into neural networks or other machine learning models. Therefore, we used it in our LSTM model.

Experiments:

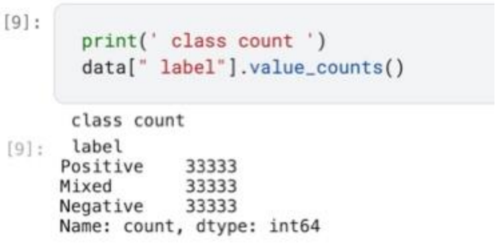
Dataset: This dataset comprises multiple datasets of reviewer reviews in Arabic for hotels, movies, products, books, and airlines. The data was obtained from Kaggle. This dataset has been gathered from diverse sources. Furthermore, it consists of two columns: the first column contains text, while the second column encompasses three categories (positive, negative, mixed). Subsequently, the text underwent cleaning, and duplicate reviewer ratings were eliminated. For data preprocessing, the data does not contain null values, and the data was balanced with 33333 samples per class. For the preprocessing of the data, we removed stop words, we did tokenization, and stemming. Details are shown below

Checking for Missing values



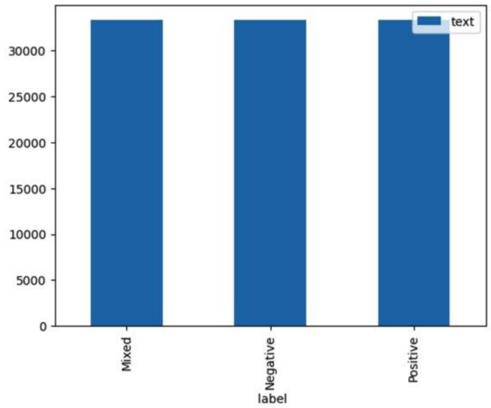
```
[8]:  print('checking for missing values')
      data.isna().any()

      checking for missing values
[8]:  text      False
       label    False
      dtype: bool
```

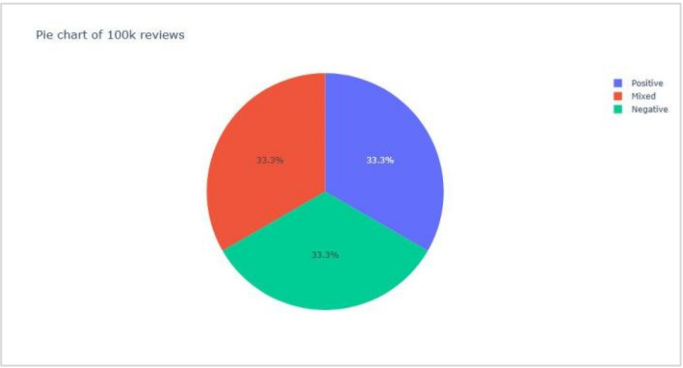
The data is balanced



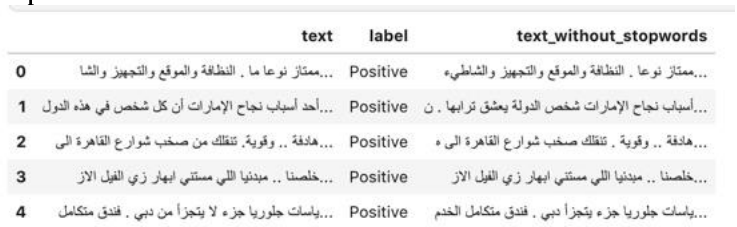
```
[9]:  print(' class count ')
      data[" label"].value_counts()

       class count
[9]:   label
      Positive    33333
      Mixed       33333
      Negative    33333
      Name: count, dtype: int64
```

distribution of sentiments



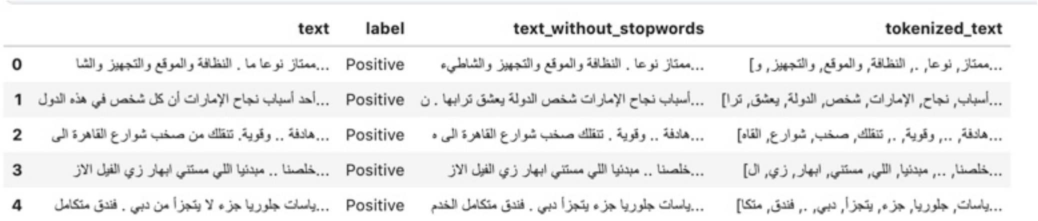


Stop word removal



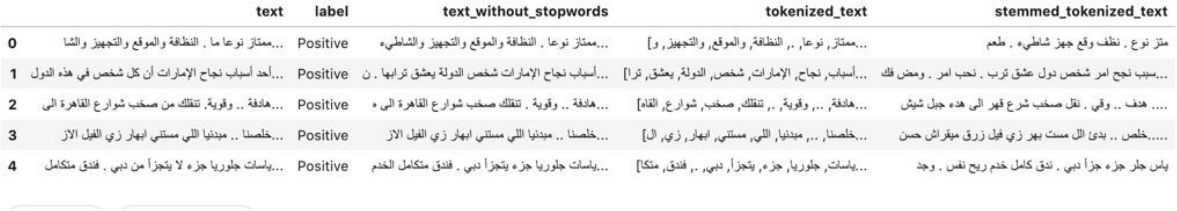
	text	label	text_without_stopwords
0	ممتاز نوعا ما . النظافة والموقع والتجهيز والشا...	Positive	ممتاز نوعا . النظافة والموقع والتجهيز والشاطيء...
1	أحد أسباب نجاح الإمارات أن كل شخص في هذه الدول...	Positive	أسباب نجاح الإمارات شخص الدولة يعشق ترابها . ن...
2	هادفة .. وقوية. تنقلك من صخب شوارع القاهرة الى...	Positive	هادفة .. وقوية . تنقلك صخب شوارع القاهرة الى ه...
3	خلصنا .. مبدنيا اللي مستني ابهار زي الفيل الاز...	Positive	خلصنا .. مبدنيا اللي مستني ابهار زي الفيل الاز...
4	ياسات جلوريا جزء لا يتجزأ من دبي . فندق متكامل...	Positive	ياسات جلوريا جزء يتجزأ دبي . فندق متكامل الخدم...

Tokenized text



	text	label	text_without_stopwords	tokenized_text
0	ممتاز نوعا ما . النظافة والموقع والتجهيز والشا...	Positive	ممتاز نوعا . النظافة والموقع والتجهيز والشاطيء...	[ممتاز, نوعا, ., النظافة, والموقع, والتجهيز, و...
1	أحد أسباب نجاح الإمارات أن كل شخص في هذه الدول...	Positive	أسباب نجاح الإمارات شخص الدولة يعشق ترابها . ن...	[أسباب, نجاح, الإمارات, شخص, الدولة, يعشق, ترا...
2	هادفة .. وقوية. تنقلك من صخب شوارع القاهرة الى...	Positive	هادفة .. وقوية . تنقلك صخب شوارع القاهرة الى ه...	[هادفة, .., وقوية, ., تنقلك, صخب, شوارع, القاه...
3	خلصنا .. مبدنيا اللي مستني ابهار زي الفيل الاز...	Positive	خلصنا .. مبدنيا اللي مستني ابهار زي الفيل الاز...	[خلصنا, .., مبدنيا, اللي, مستني, ابهار, زي, ال...
4	ياسات جلوريا جزء لا يتجزأ من دبي . فندق متكامل...	Positive	ياسات جلوريا جزء يتجزأ دبي . فندق متكامل الخدم...	[ياسات, جلوريا, جزء, يتجزأ, دبي, ., فندق, متكا...

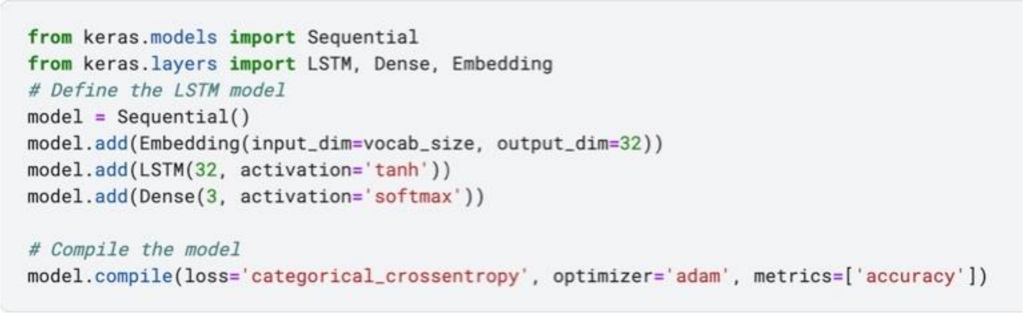
Stemming



	text	label	text_without_stopwords	tokenized_text	stemmed_tokenized_text
0	ممتاز نوعا ما . النظافة والموقع والتجهيز والشا...	Positive	ممتاز نوعا . النظافة والموقع والتجهيز والشاطيء...	[ممتاز, نوعا, ., النظافة, والموقع, والتجهيز, و...	متز نوع . نظف وقع جهز شاطيء . طعم
1	أحد أسباب نجاح الإمارات أن كل شخص في هذه الدول...	Positive	أسباب نجاح الإمارات شخص الدولة يعشق ترابها . ن...	[أسباب, نجاح, الإمارات, شخص, الدولة, يعشق, ترا...	سبب نجح امر شخص دول عشق ترب . نحب امر . ومض فك...
2	هادفة .. وقوية. تنقلك من صخب شوارع القاهرة الى...	Positive	هادفة .. وقوية . تنقلك صخب شوارع القاهرة الى ه...	[هادفة, .., وقوية, ., تنقلك, صخب, شوارع, القاه...	هدف .. وقي . نقل صخب شرع قهر الى هده جبل شيش....
3	خلصنا .. مبدنيا اللي مستني ابهار زي الفيل الاز...	Positive	خلصنا .. مبدنيا اللي مستني ابهار زي الفيل الاز...	[خلصنا, .., مبدنيا, اللي, مستني, ابهار, زي, ال...	خلص .. بدئ الل مست بهر زي فيل زرق ميقراش حسن.....
4	ياسات جلوريا جزء لا يتجزأ من دبي . فندق متكامل...	Positive	ياسات جلوريا جزء يتجزأ دبي . فندق متكامل الخدم...	[ياسات, جلوريا, جزء, يتجزأ, دبي, ., فندق, متكا...	ياس جلر جزء جزأ دبي . ندق كامل خدم ريح نفس . وجد

How you ran your experiments:

Model configuration:

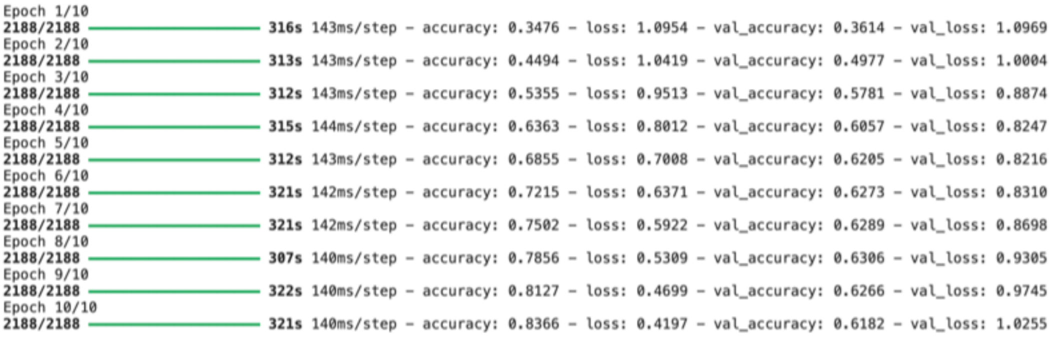


```
from keras.models import Sequential
from keras.layers import LSTM, Dense, Embedding
# Define the LSTM model
model = Sequential()
model.add(Embedding(input_dim=vocab_size, output_dim=32))
model.add(LSTM(32, activation='tanh'))
model.add(Dense(3, activation='softmax'))

# Compile the model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

1:

Learning rate	Epoch
0.01	**10**

Training and accuracy:



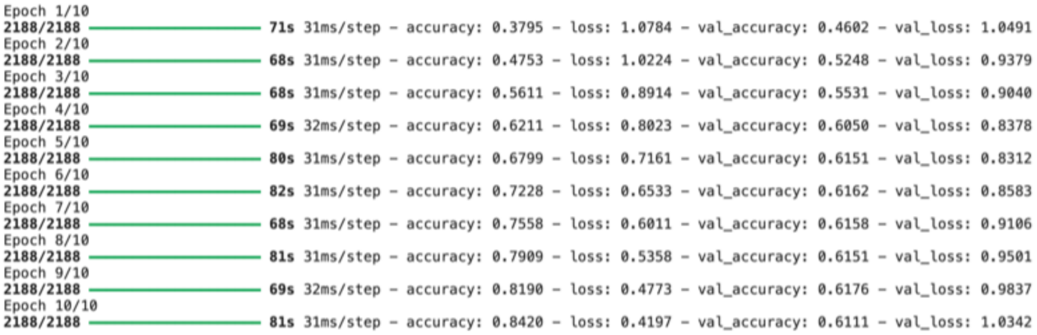
```
Epoch 1/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 316s 143ms/step - accuracy: 0.3476 - loss: 1.0954 - val_accuracy: 0.3614 - val_loss: 1.0969
Epoch 2/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 313s 143ms/step - accuracy: 0.4494 - loss: 1.0419 - val_accuracy: 0.4977 - val_loss: 1.0004
Epoch 3/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 312s 143ms/step - accuracy: 0.5355 - loss: 0.9513 - val_accuracy: 0.5781 - val_loss: 0.8874
Epoch 4/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 315s 144ms/step - accuracy: 0.6363 - loss: 0.8012 - val_accuracy: 0.6057 - val_loss: 0.8247
Epoch 5/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 312s 143ms/step - accuracy: 0.6855 - loss: 0.7008 - val_accuracy: 0.6205 - val_loss: 0.8216
Epoch 6/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 321s 142ms/step - accuracy: 0.7215 - loss: 0.6371 - val_accuracy: 0.6273 - val_loss: 0.8310
Epoch 7/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 321s 142ms/step - accuracy: 0.7502 - loss: 0.5922 - val_accuracy: 0.6289 - val_loss: 0.8698
Epoch 8/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 307s 140ms/step - accuracy: 0.7856 - loss: 0.5309 - val_accuracy: 0.6306 - val_loss: 0.9305
Epoch 9/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 322s 140ms/step - accuracy: 0.8127 - loss: 0.4699 - val_accuracy: 0.6266 - val_loss: 0.9745
Epoch 10/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 321s 140ms/step - accuracy: 0.8366 - loss: 0.4197 - val_accuracy: 0.6182 - val_loss: 1.0255
```

We noticed when epoch=10 and the learning rate=0.01 there was overfitting in the training

2:

Learning rate	Epoch
0.02	**10**

Training and accuracy:



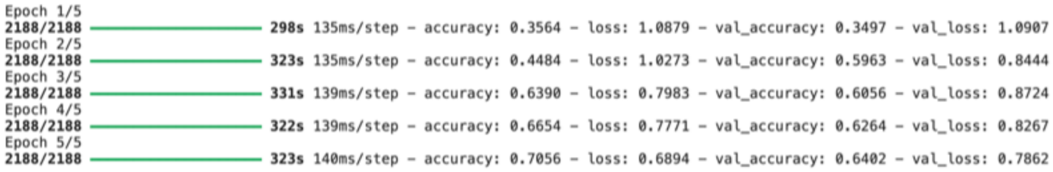
```
Epoch 1/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 71s 31ms/step - accuracy: 0.3795 - loss: 1.0784 - val_accuracy: 0.4602 - val_loss: 1.0491
Epoch 2/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 68s 31ms/step - accuracy: 0.4753 - loss: 1.0224 - val_accuracy: 0.5248 - val_loss: 0.9379
Epoch 3/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 68s 31ms/step - accuracy: 0.5611 - loss: 0.8914 - val_accuracy: 0.5531 - val_loss: 0.9040
Epoch 4/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 69s 32ms/step - accuracy: 0.6211 - loss: 0.8023 - val_accuracy: 0.6050 - val_loss: 0.8378
Epoch 5/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 80s 31ms/step - accuracy: 0.6799 - loss: 0.7161 - val_accuracy: 0.6151 - val_loss: 0.8312
Epoch 6/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 82s 31ms/step - accuracy: 0.7228 - loss: 0.6533 - val_accuracy: 0.6162 - val_loss: 0.8583
Epoch 7/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 68s 31ms/step - accuracy: 0.7558 - loss: 0.6011 - val_accuracy: 0.6158 - val_loss: 0.9106
Epoch 8/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 81s 31ms/step - accuracy: 0.7909 - loss: 0.5358 - val_accuracy: 0.6151 - val_loss: 0.9501
Epoch 9/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 69s 32ms/step - accuracy: 0.8190 - loss: 0.4773 - val_accuracy: 0.6176 - val_loss: 0.9837
Epoch 10/10
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 81s 31ms/step - accuracy: 0.8420 - loss: 0.4197 - val_accuracy: 0.6111 - val_loss: 1.0342
```

We also noticed that when learning rate=0.01 increased and when epoch=10 there was also overfitting in training.

3:

Learning rate	**Epoch**
0.02	**5**

Training and accuracy:



```
Epoch 1/5
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 298s 135ms/step - accuracy: 0.3564 - loss: 1.0879 - val_accuracy: 0.3497 - val_loss: 1.0907
Epoch 2/5
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 323s 135ms/step - accuracy: 0.4484 - loss: 1.0273 - val_accuracy: 0.5963 - val_loss: 0.8444
Epoch 3/5
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 331s 139ms/step - accuracy: 0.6390 - loss: 0.7983 - val_accuracy: 0.6056 - val_loss: 0.8724
Epoch 4/5
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 322s 139ms/step - accuracy: 0.6654 - loss: 0.7771 - val_accuracy: 0.6264 - val_loss: 0.8267
Epoch 5/5
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 323s 140ms/step - accuracy: 0.7056 - loss: 0.6894 - val_accuracy: 0.6402 - val_loss: 0.7862
```

The model still show overfitting

The evaluation metric:

The evaluation metrics used to evaluate the model performance are Accuracy, recall, and F1 score for each class. These metrics are commonly used to evaluate classification models and provide more detailed insights into model performance.

Accuracy: Measures the proportion of correctly predicted positive cases out of all predicted positive cases. It focuses on the accuracy of positive predictions.

Recall: Measures the proportion of correctly predicted positive cases out of all actual positive cases. It focuses on the model's ability to correctly identify positive cases.

F1 score: The harmonic mean of accuracy and recall. It provides a balanced measure that takes into account precision and recall.

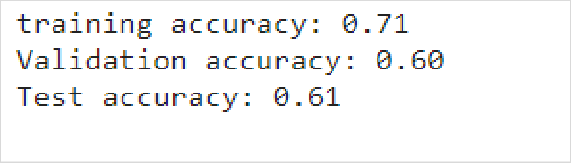
Results:

the results are shown in the table below, we faced a big challenge in preprocessing the data, also the performance is not as we expected given the size of our dataset, we expected to have high performance, but the performance was in the range 60%

using BOW:

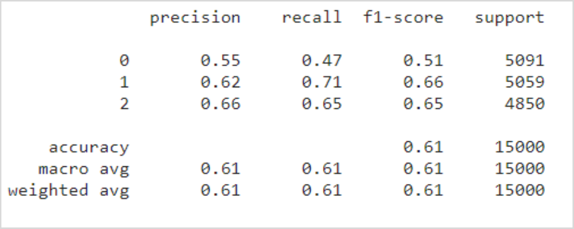
- **MultinomialNB**

Accuracy



```
training accuracy: 0.71
Validation accuracy: 0.60
Test accuracy: 0.61
```

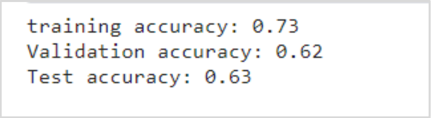
classification report



```
              precision    recall  f1-score   support

           0       0.55      0.47      0.51      5091
           1       0.62      0.71      0.66      5059
           2       0.66      0.65      0.65      4850

    accuracy                           0.61     15000
   macro avg       0.61      0.61      0.61     15000
weighted avg       0.61      0.61      0.61     15000
```

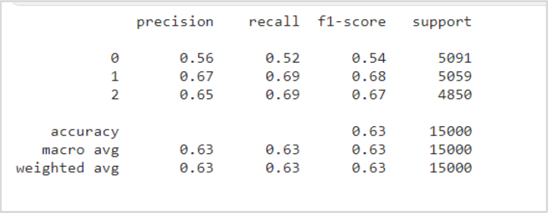
- **Logistic Regression**

Accuracy



```
training accuracy: 0.73
Validation accuracy: 0.62
Test accuracy: 0.63
```

classification report



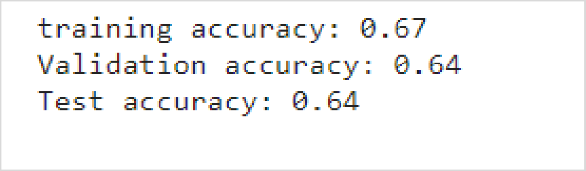
```
              precision    recall  f1-score   support

           0       0.56      0.52      0.54      5091
           1       0.67      0.69      0.68      5059
           2       0.65      0.69      0.67      4850

    accuracy                           0.63     15000
   macro avg       0.63      0.63      0.63     15000
weighted avg       0.63      0.63      0.63     15000
```

Term Frequency-Inverse Document Frequency (TF-IDF)

These feature names correspond to the columns of the TF-IDF matrix, where each feature name represents a unique word in the corpus. 99999 rows and 84476 columns. This typically indicates that you have 99999 samples.

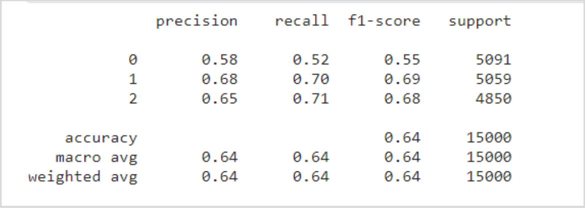
- **Logistic Regression :**

Accuracy



```
training accuracy: 0.67
Validation accuracy: 0.64
Test accuracy: 0.64
```

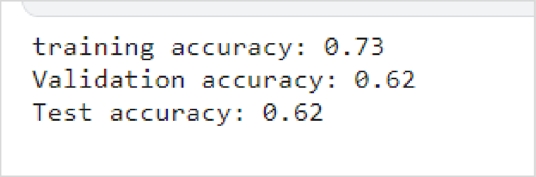
classification report



```
              precision    recall  f1-score   support

           0       0.58      0.52      0.55      5091
           1       0.68      0.70      0.69      5059
           2       0.65      0.71      0.68      4850

    accuracy                           0.64     15000
   macro avg       0.64      0.64      0.64     15000
weighted avg       0.64      0.64      0.64     15000
```

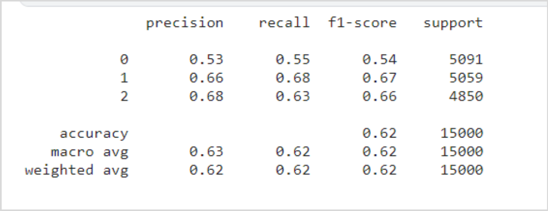
- **MultinomialNB**:

Accuracy



```
training accuracy: 0.73
Validation accuracy: 0.62
Test accuracy: 0.62
```

classification report



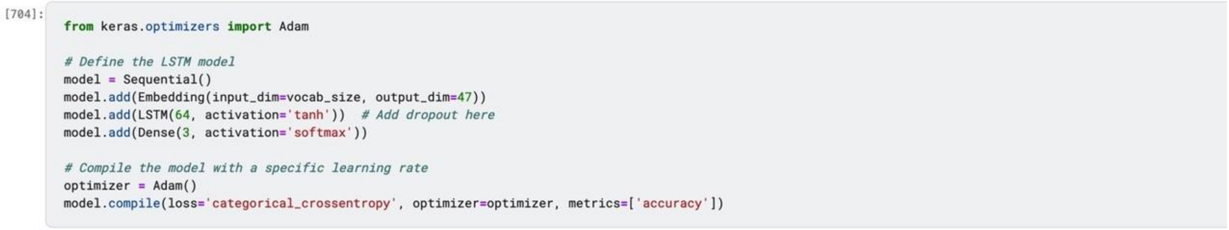
```
              precision    recall  f1-score   support

           0       0.53      0.55      0.54      5091
           1       0.66      0.68      0.67      5059
           2       0.68      0.63      0.66      4850

    accuracy                           0.62     15000
   macro avg       0.63      0.62      0.62     15000
weighted avg       0.62      0.62      0.62     15000
```

for TF-IDF logistic regression also have better performance with 64 accuracy and MultinomialNB with 62 accuracy

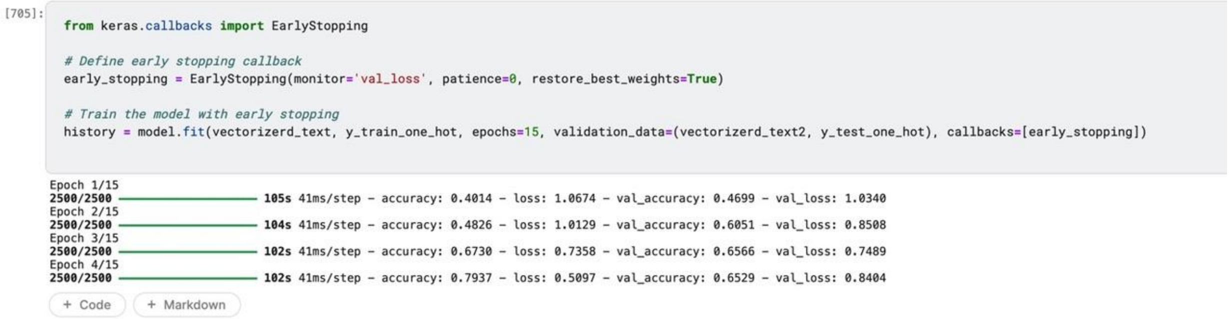
IN GENERAL , Logistic regression tends to perform much better than MultinomialNB

LSTM:

model

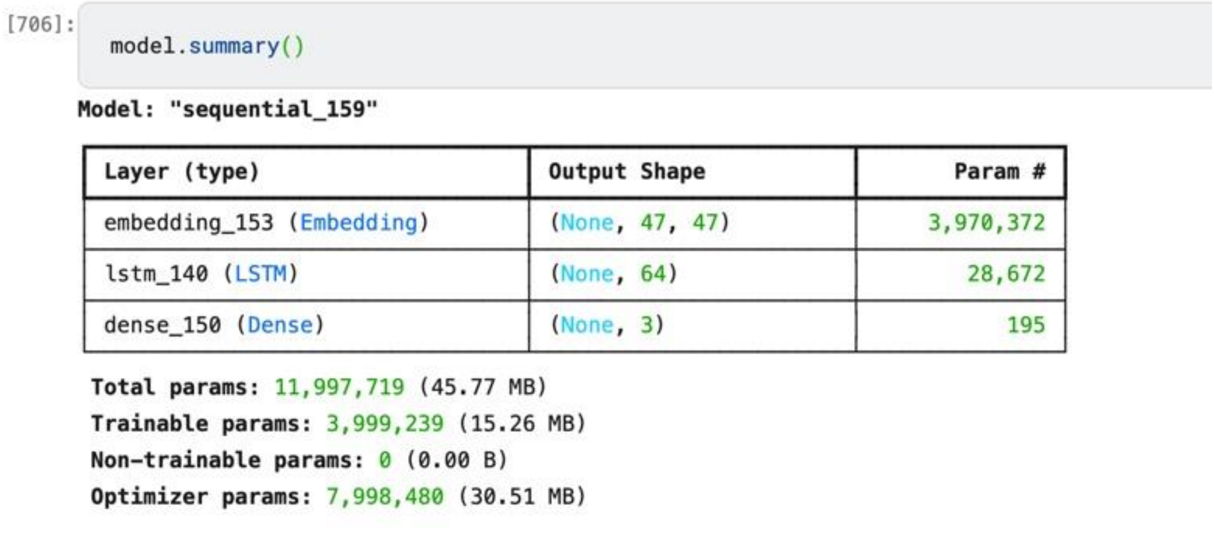


```
[704]:
from keras.optimizers import Adam

# Define the LSTM model
model = Sequential()
model.add(Embedding(input_dim=vocab_size, output_dim=47))
model.add(LSTM(64, activation='tanh'))  # Add dropout here
model.add(Dense(3, activation='softmax'))

# Compile the model with a specific learning rate
optimizer = Adam()
model.compile(loss='categorical_crossentropy', optimizer=optimizer, metrics=['accuracy'])
```



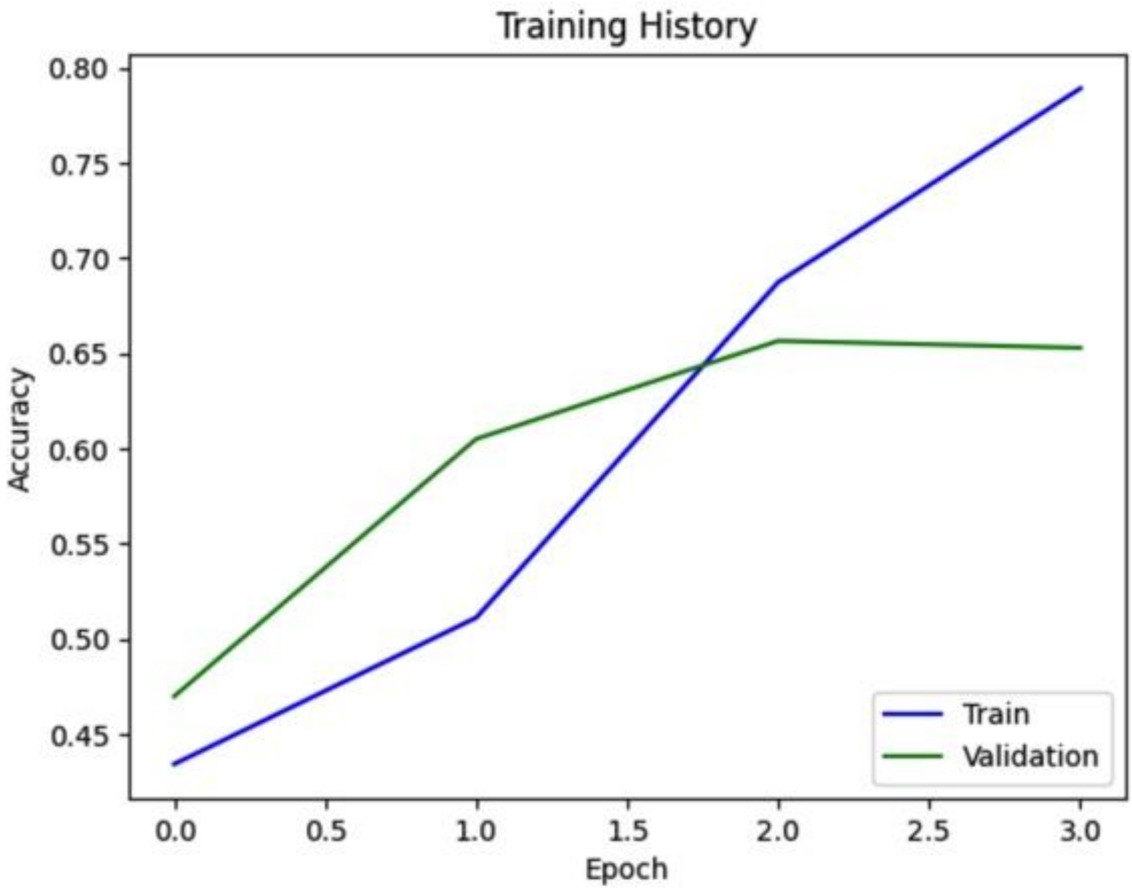
```
[705]:
from keras.callbacks import EarlyStopping

# Define early stopping callback
early_stopping = EarlyStopping(monitor='val_loss', patience=0, restore_best_weights=True)

# Train the model with early stopping
history = model.fit(vectorizerd_text, y_train_one_hot, epochs=15, validation_data=(vectorizerd_text2, y_test_one_hot), callbacks=[early_stopping])
```

```
Epoch 1/15
2500/2500 ━━━━━━━━━━━━━━━━━━━━ 105s 41ms/step - accuracy: 0.4014 - loss: 1.0674 - val_accuracy: 0.4699 - val_loss: 1.0340
Epoch 2/15
2500/2500 ━━━━━━━━━━━━━━━━━━━━ 104s 41ms/step - accuracy: 0.4826 - loss: 1.0129 - val_accuracy: 0.6051 - val_loss: 0.8508
Epoch 3/15
2500/2500 ━━━━━━━━━━━━━━━━━━━━ 102s 41ms/step - accuracy: 0.6730 - loss: 0.7358 - val_accuracy: 0.6566 - val_loss: 0.7489
Epoch 4/15
2500/2500 ━━━━━━━━━━━━━━━━━━━━ 102s 41ms/step - accuracy: 0.7937 - loss: 0.5097 - val_accuracy: 0.6529 - val_loss: 0.8404
```

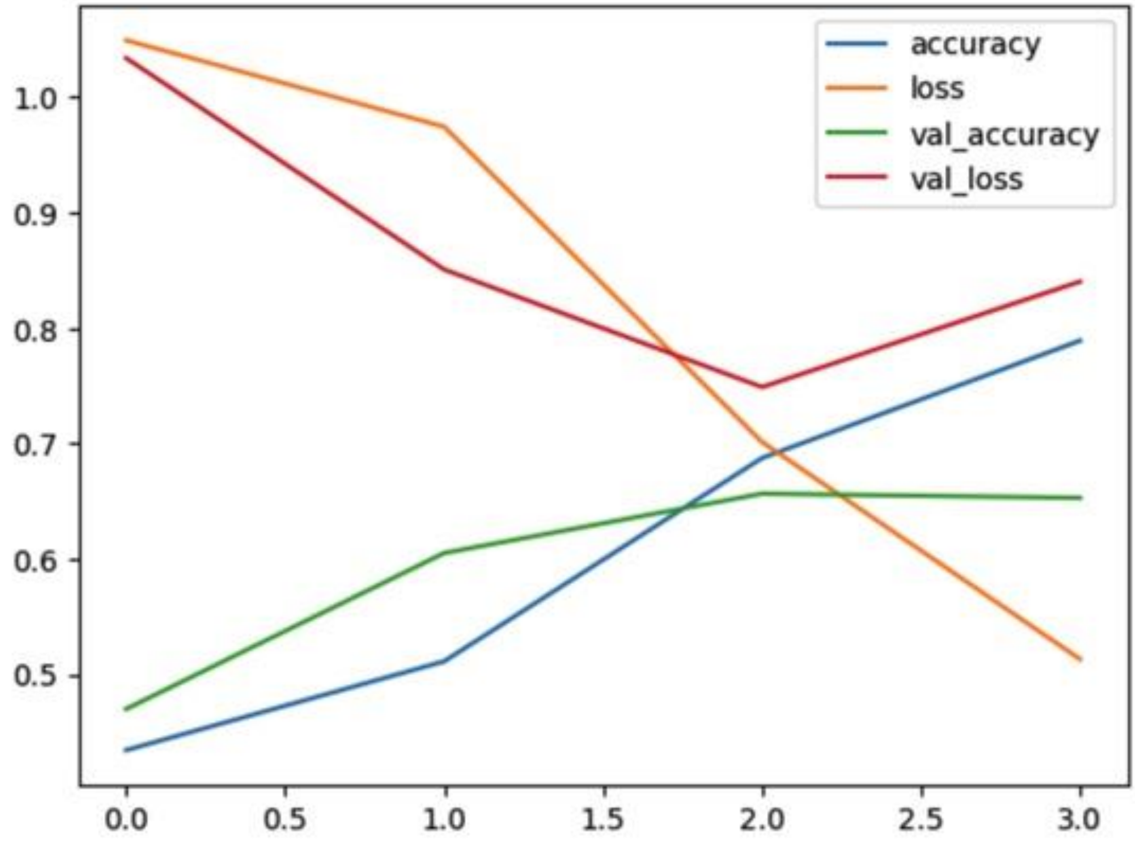


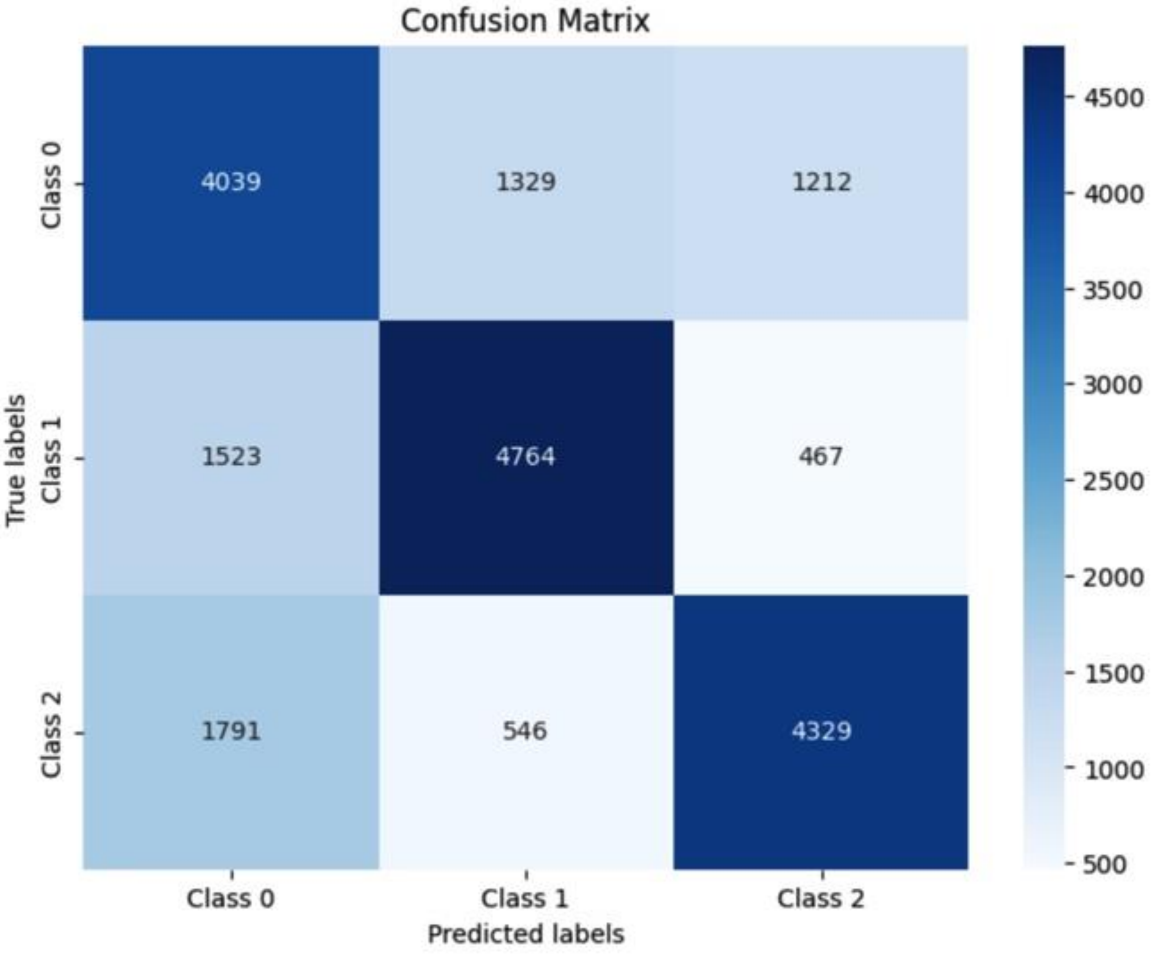
```
[706]:
model.summary()
```

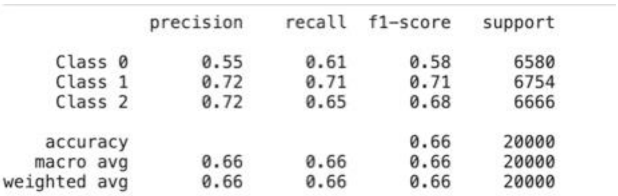
```
Model: "sequential_159"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ embedding_153 (Embedding)       │ (None, 47, 47)         │     3,970,372 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ lstm_140 (LSTM)                 │ (None, 64)             │        28,672 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_150 (Dense)               │ (None, 3)              │           195 │
└─────────────────────────────────┴────────────────────────┴───────────────┘
 Total params: 11,997,719 (45.77 MB)
 Trainable params: 3,999,239 (15.26 MB)
 Non-trainable params: 0 (0.00 B)
 Optimizer params: 7,998,480 (30.51 MB)
```


Training History
Accuracy
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0.45
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Epoch
Train
Validation

```
<Figure size 800x500 with 0 Axes>
```


accuracy
loss
val_accuracy
val_loss
1.0
0.9
0.8
0.7
0.6
0.5
0.0
0.5
1.0
1.5
2.0
2.5
3.0





```
              precision    recall  f1-score   support

     Class 0       0.55      0.61      0.58      6580
     Class 1       0.72      0.71      0.71      6754
     Class 2       0.72      0.65      0.68      6666

    accuracy                           0.66     20000
   macro avg       0.66      0.66      0.66     20000
weighted avg       0.66      0.66      0.66     20000
```

This is our latest experience, so we did not specify the batch size to be used in the training process, but we decided to use the default batch size. Further, we observed that the model was generally overfit and, as a consequence, we decided to use an Early Stop strategy.

Comparison between the statistical- based and deep learning-based methods

Statistical methods:
Two algorithms were applied in our project: first, the logistic regression algorithm, which used for classification, and second, the Naive Bayes algorithm, which selects words based on probabilities, so that the algorithms provide good performance when the data is linearly separable, so it is considered easier and simpler than deep learning methods, which are scalable. For interpretation and gives a clear understanding of how to predict.

-Features (words):
- BOW was used to search for the text as a group of individual words so that each word is considered as a feature. In addition to that, the accuracy of the results obtained in logistic regression was 63% and 61% MultinomialNB.

•TF-IDF was used to extract information from the text and analyze the text. The accuracy of the results in logistic regression is 64% and MultinomialNB is 62%.

Deep learning:
The LSTM model was applied in our project, and it was noted that it gives better performance in complex patterns, so it can outperform statistical methods when the data set is large, but it is considered a more complex model because it is less interpretable, which makes it more difficult to interpret the model.

-Features (words):

•TextVectorization is performed to convert text into strings of integers, as the accuracy of the results obtained by LSTM shows is 66%.

Models :	Logistic regression & **MultinomialNB**	LSTM
Features:	BoW TF-IDF	Text vectorization, Word Embedding
Performance	Both models showed similar results thatvary between 61% to 63% logistic regression perform better than **MultinomialNB**	The model showed higher accuracy than machine learning models with accuracy =64% although its not very high but mainly it is because of the Arabic text which is hard to deal with.

Conclusion:

In conclusion, this project sheds light on the challenges and approaches in sentiment analysis of Arabic language reviews using natural language processing techniques. By using the "Arabic 100k Reviews" dataset, valuable insights were gained into understanding and analyzing Arabic texts. The project demonstrated the effectiveness of machine learning models such as Logistic Regression and MultinomialNB in classifying Arabic reviews. Specifically, the Logistic Regression model outperformed the MultinomialNB model in terms of accuracy, indicating its ability to effectively represent and classify Arabic texts in the used dataset. Furthermore, the use of the LSTM model in deep learning showed significant capabilities in capturing context and sequence in Arabic texts. This highlights the power of deep learning models in leveraging sequential structures and analyzing context in natural language processing tasks.

References:

[1] Aliman, G., Nivera, T. F. S., Olazo, J. C. A., Ramos, D. J. P., Sanchez, C. D. B., Amado, T. M., ... & Valenzuela, I. C. (2022). Sentiment analysis using logistic regression. *Journal of Computational Innovations and Engineering Applications*, *7*(1), 35-40.

[2] Murthy, G. S. N., Allu, S. R., Andhavarapu, B., Bagadi, M., & Belusonti, M. (2020). Text based sentiment analysis using LSTM. *Int. J. Eng. Res. Tech. Res*, *9*(05).

[3] Kasri, M., Birjali, M., & Beni-Hssane, A. (2019, October). A comparison of features extraction methods for Arabic sentiment analysis. In Proceedings of the 4th international conference on big data and internet of things (pp. 1-6)

[4] Arabic 100k Reviews. (2020, March 7). Kaggle. https://www.kaggle.com/datasets/abedkhooli/arabic-100k-reviews

[5] O. (2024, January 30). ArabicSentimentAnalysis. Kaggle. https://www.kaggle.com/code/omarmagdy33/arabicsentimentanalysis

[6] Susanti, A. R., Djatna, T., & Kusuma, W. A. (2017). Twitter's sentiment analysis on GSM services using Multinomial Naïve Bayes. Telkomnika (Telecommunication Computing Electronics and Control), 15(3), 1354-1361.

Team Contribution:

We as a team, worked in close collaboration and with a unified team spirit to achieve our common goal. We divided the tasks among ourselves, but we also worked together and supported each other in the process of completing the tasks. We checked and corrected each other's work and wrote the content collaboratively.