INTRODUCTION

This report is aimed at analysing a comprehensive NLP task application of a text classification on Amazon online review. Natural Language Processing is a theoretically motivated range of computational techniques for analyzing and representing naturally occurring texts at one or more levels of linguistic analysis to achieve human-like-language processing for a range of tasks or applications. (Liddy,2001)

IMPORTANCE OF NLP APPLICATION

NLP bridges the gap in human-machine communication by enabling computers to comprehend and process human language, allowing for more natural interactions. Applications such as chatbots, machine translation tools, and virtual assistants (like Siri and Alexa) exhibit this.

NLP assists in deriving meaning from this data, facilitating applications such as information retrieval (locating pertinent documents based on searches), topic modelling (identifying important topics in texts), and sentiment analysis (understanding consumer opinions).

Natural Language Processing (NLP) enables sentiment analysis, enabling organizations to assess customer opinions from various sources such as social media, reviews, and polls. It facilitates interactions with chatbots, which automate responses and enhance user experiences.

In addition, Amazon, as an e-commerce site, employs a classification system to organize products into many pertinent categories such as Groceries, Kitchen, and Electronics. This classification enhances user experience by helping users in locating their desired items more effortlessly.

Also, classification helps in fraudulent transaction detection because trained models learn patterns of suspicious and fraudulent act to flag potentially fraudulent order

Amazon and other e-commerce platforms can gain insights into consumer satisfaction with their products by categorizing reviews as positive, negative, or neutral.

BACKGROUND REVIEW

Several studies have researched sentiment analysis and text classification. Murphy et al, (2020). Proposed a sentiment classification approach based on LSTM for text data analysis. The approach aimed to automatically identify the polarity of reviews as positive or negative sentiments. The experimental study employed the IMDB and Amazon Product datasets for sentiment analysis. LSTM networks demonstrated better performance in sentiment classification tasks with an accuracy of 85% when trained on large datasets. However, the application was found to be very good based on the model accuracy of 85%.

SMART OBJECTIVES.

- **Specific:** Develop and implement a traditional and deep learning machine based on a sentiment analysis model to classify customer reviews from the Amazon Review dataset as Negative and positive sentiments.
- **Measurable**: Achieve a minimum accuracy of 90% in sentiment classification using the dataset.
- **Achievable**: Employ the existing knowledge of traditional and deep learning models and sentiment analysis techniques to train the model.
- **Relevant**: The objective is in line with the significance of sentiment analysis in comprehending customer opinions and can offer significant insight for product development and marketing strategies.
- **Time-bound**: Ensure the timely model development, training, and evaluation.

COMMENT ON THE DATASET

Amazon review dataset was loaded and read in CSV format. The dataset has a size of 19996 and it consists of 19996 rows and 2 columns. The dataset was well structured. The value counts calculate the number of times each unique value appears within that column, from the label column, 1 is 15230 while 0 is 4766 which reveals that the data is not balanced.

Suitability of the Dataset.

The dataset is suitable for the problem because of its pertinence in targeting sentiment analysis as it encompasses expressions of opinions and evaluations on products. The actual data shows how individuals express positive, negative, and neutral sentiments through language. Also, the size and the diversity of the dataset are suitable for deep learning techniques because large data model performs better than large datasets.

Strengths.

- The dataset is relevant for sentiment analysis.
- Amazon offers a huge database of reviews in many different product categories.
- The datasets are pre-labeled which is essential for supervised learning using LSTM.

Weakness.

- Informal languages like emojis and slang are frequently found in reviews, hence some preprocessing steps might be needed for the model to handle these elements effectively and efficiently.
- The reviews may contain some sarcasm which may not reflect clear-cut sentiment.
- The dataset may be biased due to the review which might be skewed towards a particular product.

EXPLORATORY ANALYSIS AND DATA PREPROCESSING

A function was defined to clean the data sequentially to make it suitable for sentiment analysis. **Convert text to lowercase**: The text was converted to lowercase to ensure consistency as sometimes sentiment can be expressed through capital letters.

Remove punctuation: Punctuation marks like commas, exclamation points, and periods were removed to make the text easier to understand for the model as they do not significantly add to sentiment meaning.

Tokenization of text: Text was split into meaningful tokens or individual words using nltk.word_tokenize function.

Stop word: Stop words were removed to minimize noise and enable the model to focus on the more content-dense words. Stop words have minimal impact on sentiment meaning.

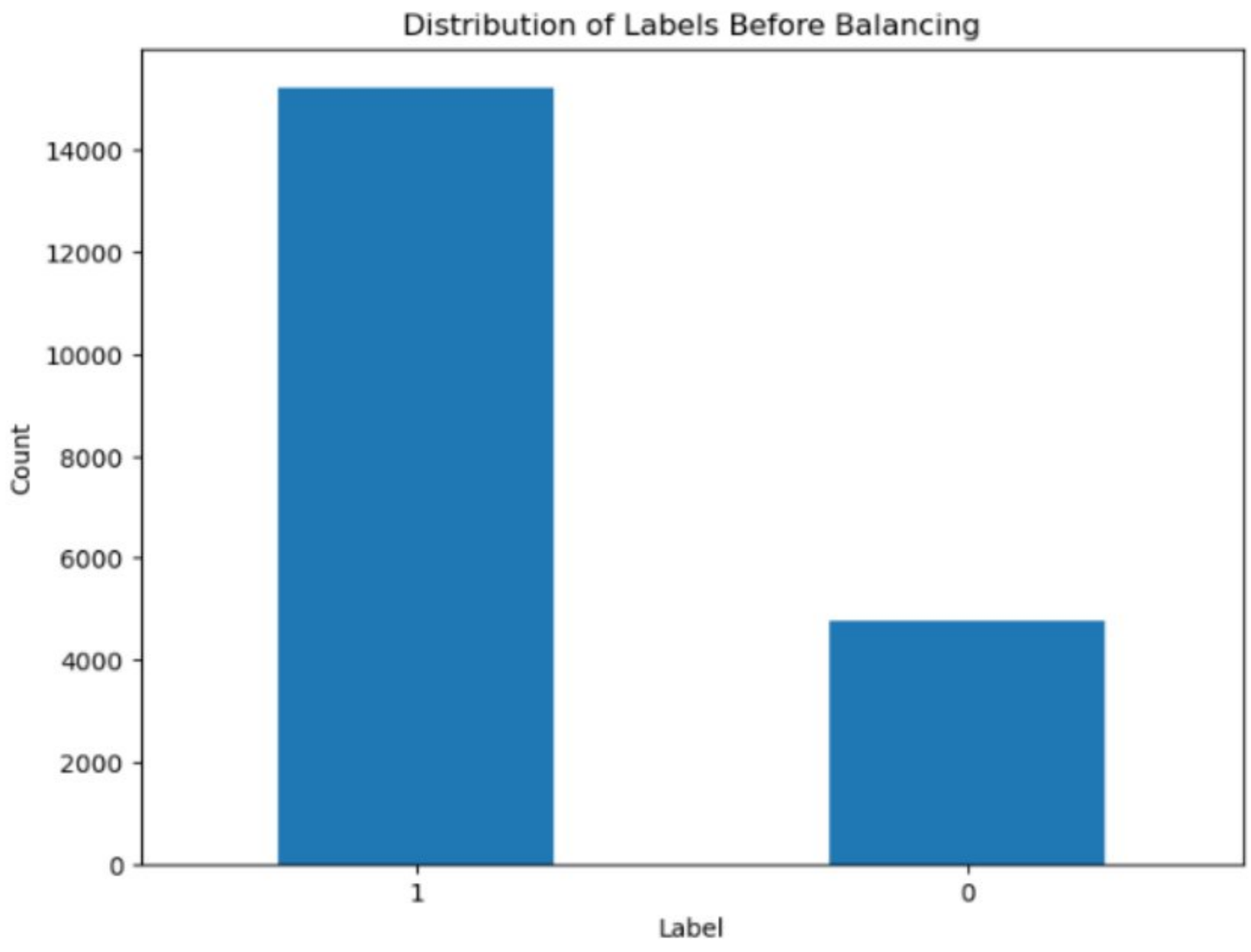

Fig.1 Distribution of Labels Before Balancing

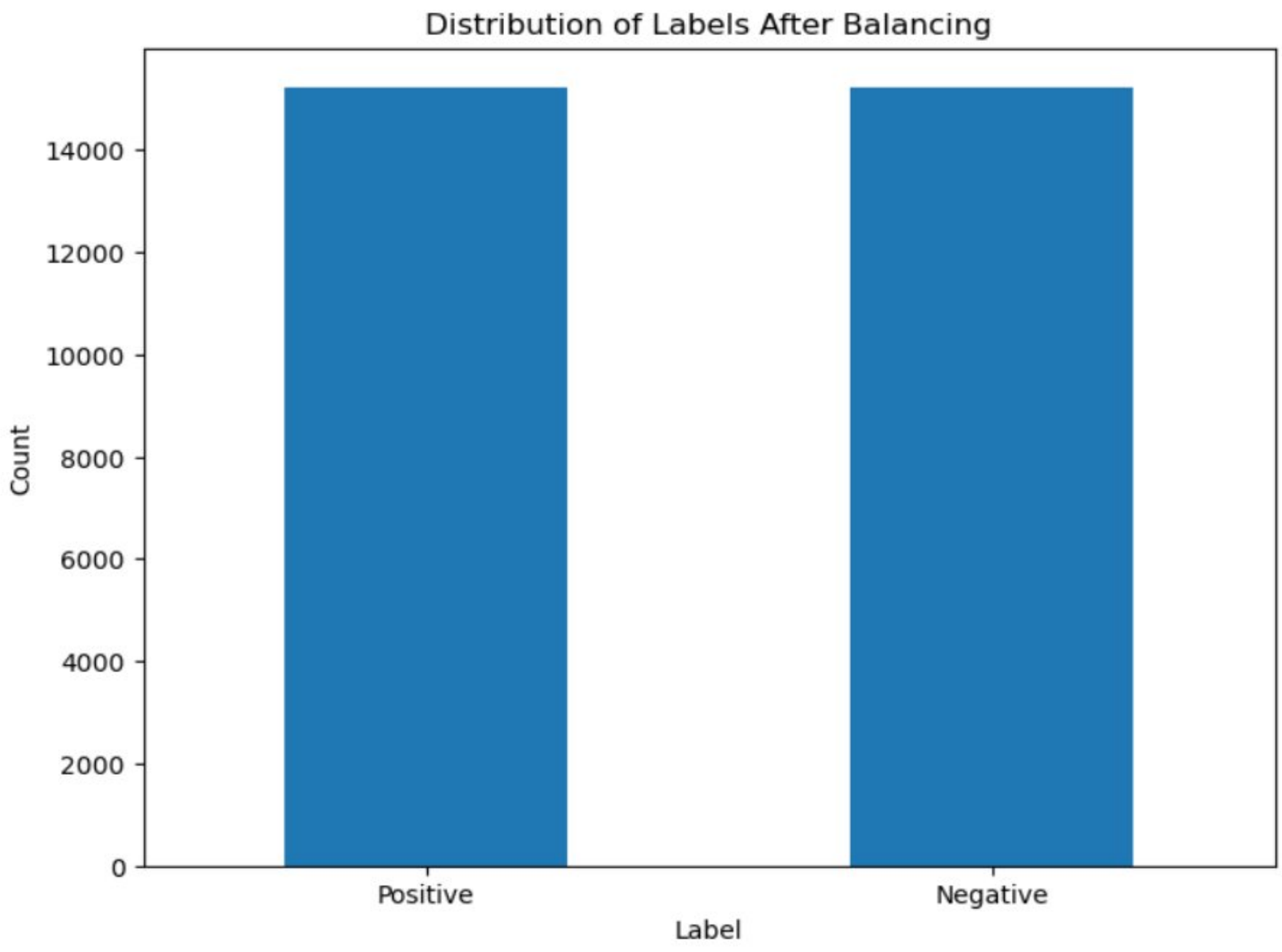

Fig.2. Distribution of Labels After Balancing

Fig.1 shows the visualization of the distribution of labels before balancing, while Fig. 2. shows the distribution of labels after balancing. The dataset exhibits a class imbalance, with the negative sentiment class having a lower class than the positive sentiment class. This can cause the negative sentiment class, which is the minority class, to perform poorly in the model. To establish a more balanced dataset, the imbalance problem was resolved by using oversampling techniques, which involved randomly duplicating cases from the negative sentiment class.

COMPARE A RANGE OF TRADITIONAL MACHINE LEARNING.

There are several ranges of traditional machine-learning methods which are suitable and effective for sentiment analysis.

Logistic Regression: is a statistical model applied to tasks involving binary classification like positive and negative sentiments. However, logistic regression may not handle complex sentiment patterns because it is limited to linear relationships.

Naïve Bayes: This methods is easy to implement and effective for large datasets and suitable for sparse feature. Its drawback is it may not capture complex wok because it assumes that features are independent of one another.

Support Vector Machines This method is effective for non-linear interactions involving the kernel function and also strong large-scale data.

Naïve Bayes and Logistic Regression of traditional machine learning were employed for this NLP application tasks.

COMPARE A RANGE OF DEEP LEARNING METHOD.

Recurrent Neural Networks (RNNs): According to Ruales, (2011). Recurrent Neural Networks are types of neural networks that contain directed loops. This loop represents the propagation of activations of future inputs in a sequence**.** Recurrent Neural Networks (RNNs) have demonstrated success in various natural language processing tasks, surpassing traditional n-gram models, and achieving state-of-the-art performance in speech recognition tasks. (Mikolov et al. 2010).

Long Short-Term Memory (LSTM): Wang et al (2015) in their research highlight the effectiveness of LSTM recurrent networks in sentiment prediction tasks, showcasing improved performance compared to traditional feature-engineering approaches and non-neural models. The LSTM model's ability to capture interactions between words and handle special linguistic functions like negation signifies its potential for enhancing sentiment analysis in text data processing. However, LSTM can be computationally expensive to train therefore it requires hyperparameter Tuning.

Convolutional Neural Networks: A Convolutional Neural Network is a special type of deep learning architecture designed for processing and analyzing spatial data such as video and images. CNN works systematically by processing input data through a series of layers and extracting features from the raw input data. proposes an approach for sentiment analysis of Twitter data using deep learning techniques, specifically focusing on the application of Convolutional Neural Networks (CNN) for improved accuracy in sentiment classification tasks. The effectiveness of CNNs in sentiment analysis tasks, particularly in the context of Twitter data, showcasing improved accuracy and performance compared to traditional machine learning methods. The adoption of deep learning techniques, specifically CNNs, highlights the potential for enhancing sentiment analysis in text data processing applications. (Wang et al 2017).

IMPLEMENTATION OF TRADITIONAL AND DEEP LEARNING MODELS

Naïve Bayes was employed and the model was trained without using TF-IDF and also trained with TF-IDF to compare the performance and the accuracy by splitting the data into training and testing sets using the preprocessed Amazon resampled data frame. When data is split it is to ensure that the model doesn't memorize the training data and can generalize well to unseen examples during the evaluation process. After that, the model was initialized by importing sklearn,feature_extraction.text library. Text Vectorization was applied to fit and transform the data text data into a numerical format that traditional machine learning models can understand and process. The trained Naïve Bayes classifier predicted sentiment labels and a classification report and confusion matrix were generated.

Logistic Regression: This a commonly employed classification method that is particularly well suited for applications such as sentiment analysis with the aim of predicting a binary

outcome like positive and negative. The logistic was initialized by importing LogisticRegressionfrom sklearn.linear_model library The model was trained using fit and also vectorized with TF-IDF and without TF-IDF to compare the results. Prediction on test sets was made using the trained Logistic Regression Classifier to predict sentiment labels. Classification reports and confusion matrix were generated for both the model with TF-IDF and the model without TF-IDF to show the appropriate metric evaluation and to understand how the performance of the model is in classifying the classes of sentiments as positive and negative.

Recurrent Neural Network (RNN):

- **Text Tokenization:** The text data was tokenized **by** converting text data to sequences of integers.
- **Padding Sequence:** sequences were padded to ensure uniform length.
- **Label Encoding:** sentiment text was converted to numerical values for the model.
- **Data split:** Splitting the data into training and testing sets to train and evaluate the model.
- **Model Architecture:** A sequential model with 3 layers was defined, embedding (input_dim = vocab_size and output_dim+embedding_dim), Simple RNN layer (units= 64, dropout= 0.2), and Dense layer which has the (activation function 'sigmoid')
- **Model Compilation:** Optimizer = adam. Loss= 'binary_crossentropy' and metric = accuracy.
- **Model Training:** The RNN model was trained on training data (X_train, y_train) with epoch = 5, batch size = 32, and validation split=0.1.
- **Model Evaluation:** The trained model was evaluated on the test data(X_test,y_test) and the loss and accuracy test was printed
- **Predictions:** The test sets were predicted using a threshold of 0.5.
- **Classification Report:** To check the model's performance using the following metrics, precision, recall, and F1-score for positive and negative sentiments.

Recurrent Neural Network (RNN) with TF-IDF

- **Data split:** Splitting the data into training and testing sets to train and evaluate the model (80% training, 20% testing).
- **Text Vectorization with TF-IDF**: Text data was converted to numerical values using Tfidfvectorizer. The maximum features used was 10,000. Fitted vectorizer was used to transform the training and testing data sets and the data was reshaped to fit the RNN model.
- **Model Architecture:** A sequential layer was defined with Tensorflow's MirroredStrategy for potential GPU distribution. Simple RNN layer (units= 128, activation function = relu, dropout= 0.2), and Dense layer which has the (units =1,activation = 'sigmoid')

- **Model Compilation:** Optimizer = adam. With learning rate of 0.01 Loss = 'binary_crossentropy' for binary classification, and metric = accuracy
- **Early Stopping:** Establishes an EarlyStopping call back to track validation loss and terminate training after three epochs (patience) if it doesn't get better.
- **Model Training:** Train the model with a batch size of 64, epochs =50, and validation split of 10% to monitor the performance and prevent overfitting.
- **Model Evaluation:** The trained model was evaluated on the test data(X_test_rnn, y_test) and the loss and accuracy test was printed
- **Predictions:** The test sets were predicted using a threshold of 0.5.
- **Classification Report:** To check the model's performance using the following metrics, precision, recall, and F1-score for positive and negative sentiments.

Recall that activation function like relu is a regularization technique.

Long Short-Term Memory (LSTM) Model without TF-IDF

- **Text Tokenization:** The text data was tokenized **by** converting text data to sequences of integers.
- **Padding Sequence:** sequences were padded to ensure uniform length for the LSTM model.
- **Label Encoding:** sentiment text was converted to numerical values for the model.
- **Data split:** Splitting the data into training and testing sets to train and evaluate the LSTM model.
- **Model Architecture:** A sequential model with 3 layers was defined, embedding (input_dim = vocab_size and output_dim+embedding_dim), Simple RNN layer (units= 64, dropout= 0.2), and Dense layer which has the (activation function 'sigmoid')
- **Model Compilation:** Optimizer = adam. Loss= 'binary_crossentropy' and metric = accuracy.
- **Model Training:** The RNN model was trained on training data (X_train, y_train) with epoch = 5, batch size = 32, and validation split=0.1.
- **Model Evaluation:** The trained LSTM model was evaluated on the test data(X_test,y_test) and the loss and accuracy test was printed
- **Predictions:** The test sets were predicted using a threshold of 0.5.
- **Classification Report:** To check the model's performance using the following metrics, precision, recall, and F1-score for positive and negative sentiments.

Long Short_ Term Memory (LSTM) Model using TF-IDF

- **Data split:** Splitting the data into training and testing sets to train and evaluate the model (80% training, 20% testing).

- **Text Vectorization with TF-IDF**: Text data was converted to numerical values using Tfidfvectorizer. The maximum features used was 10,000. Fitted vectorizer was used to transform the training and testing data sets and the data was reshaped to fit the RNN model.
- **Model Architecture:** A sequential layer was defined with Tensorflow's MirroredStrategy for potential GPU distribution. Simple RNN layer (units= 128, activation function = relu, dropout= 0.2), and Dense layer which has the (units =1,activation = 'sigmoid')
- **Model Compilation:** Optimizer = adam. With learning rate of 0.01 Loss = 'binary_crossentropy' for binary classification, and metric = accuracy
- **Early Stopping:** Establishes an EarlyStopping call back to track validation loss and terminate training after three epochs (patience) if it doesn't get better.
- **Model Training:** Train the model with a batch size of 64, epochs =50, and validation split of 10% to monitor the performance and prevent overfitting.
- **Model Evaluation:** The trained model was evaluated on the test data(X_test_rnn, y_test) and the loss and accuracy test was printed
- **Predictions:** The test sets were predicted using a threshold of 0.5.
- **Classification Report:** To check the model's performance using the following metrics, precision, recall, and F1-score for positive and negative sentiments.

HYPERPARAMETER TUNING

Model	Learning Rate	Batch size	Test Accuracy
RNN Model	**0.001**	**128**	**0.7555**
LSTM Model	**0.001**	**128**	**0.9007**

Table 1

The RNN and LSTM model's performance with hyperparameter tuning did not improve compared to the previous model.

EVALUATION

Naïve Bayes Method

Classification Report:

	precision	recall	f1-score	support
0	0.89	0.93	0.91	3021
1	0.93	0.88	0.91	3071
accuracy			0.91	6092
macro avg	0.91	0.91	0.91	6092
weighted avg	0.91	0.91	0.91	6092

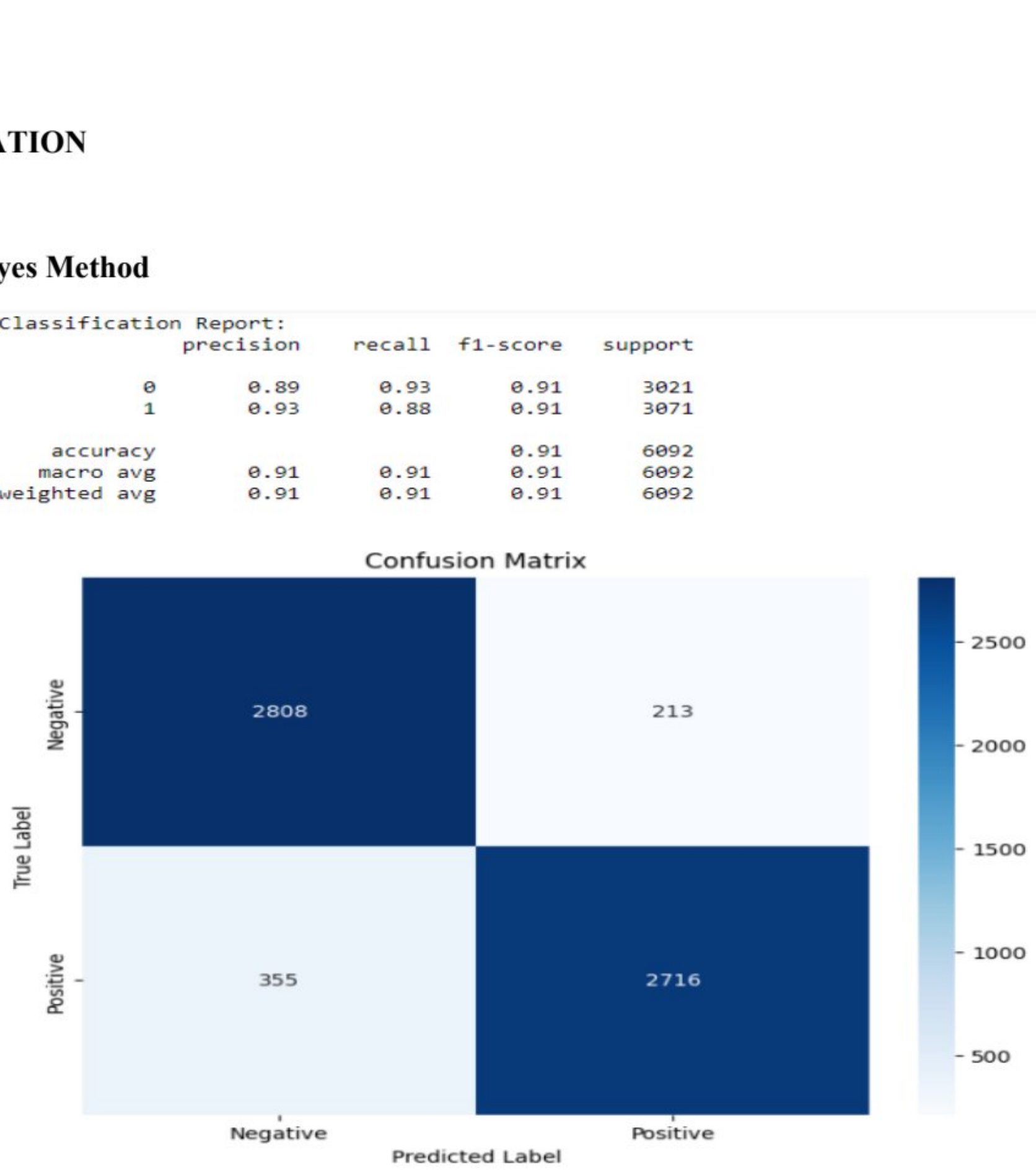

Fig.3. Classification Report and a Confusion Matrix for Naïve Bayes Classifier

```
Classification Report (Naive Bayes with TF-IDF):
              precision    recall  f1-score   support

           0       0.96      0.13      0.23       978
           1       0.78      1.00      0.88      3022

    accuracy                           0.79      4000
   macro avg       0.87      0.56      0.55      4000
weighted avg       0.82      0.79      0.72      4000

Confusion Matrix (Naive Bayes with TF-IDF):
[[ 129  849]
 [   6 3016]]
```

Fig.4. Classification Report for Naïve Bayes TF-IDF

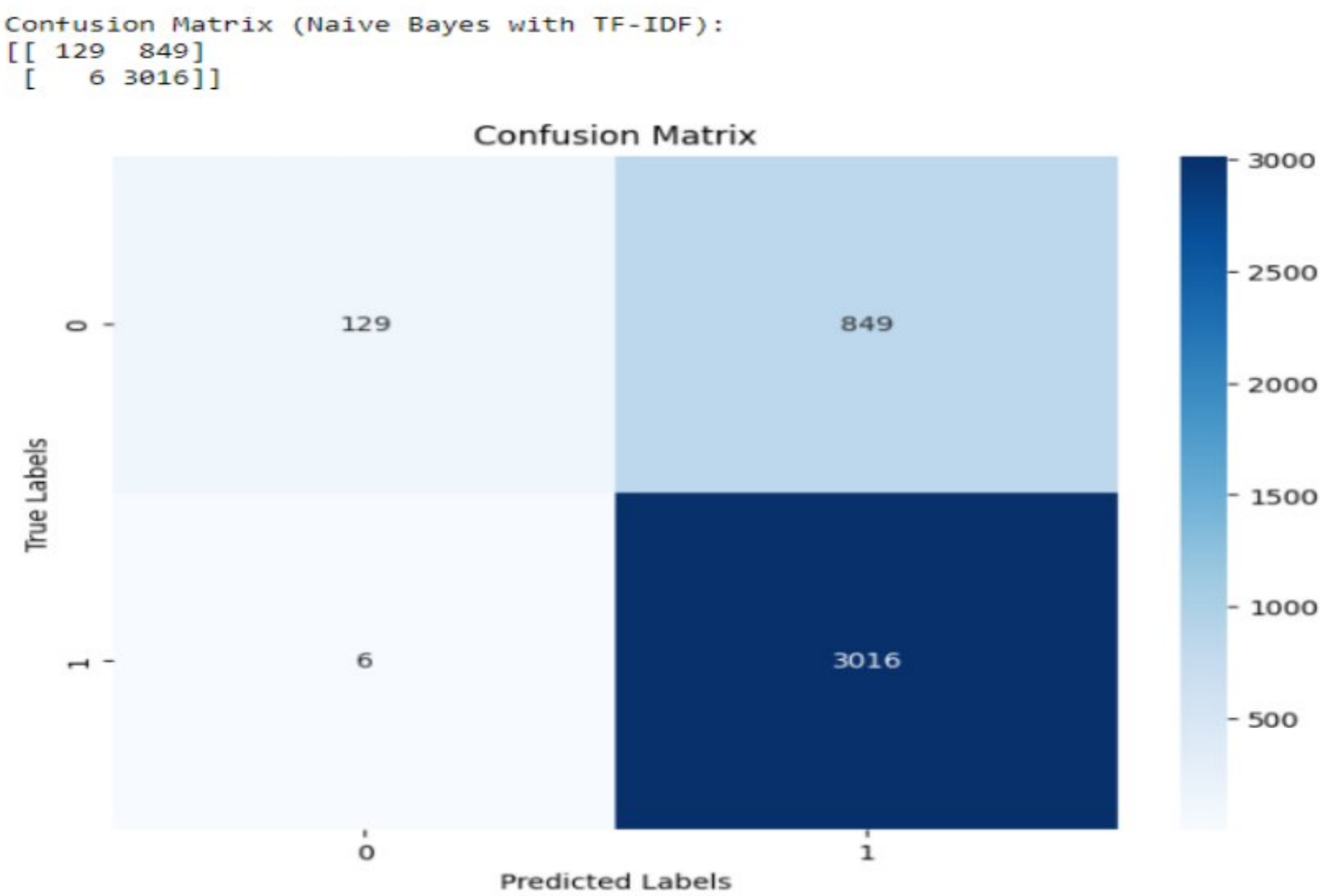

Fig.5. Confusion Matrix for Naïve Bayes TF-IDF

Fig.3., 4 and 5 above is a confusion matrix and classification report for a sentiment analysis model using Naïve Bayes Classifier and TF-IDF to evaluate the performance of the model on classifying reviews as positive and negative

Confusion Matrix: We have the True Label and the Predicted label these are rows and columns respectively The True label depicts the actual sentiment labels and shows Positive and Negative. On the other hand, the Predicted Label depicts the sentiment label the model was able to predict. Fig.3 the model correctly classified 2500 positive reviews, predicted positive and positive but incorrectly classified 2808 negative reviews as positive, predicting positive but negative. The model was able to classify and predict 2716 negative reviews

correctly while 355 positive reviews were incorrectly classified and predicted, actually negative but predicted positive. Fig.5 The model incorrectly predicted 849 negative and they are actually positive and correctly predicted 129 negative. 6 samples were incorrectly predicted positive and correctly predicted 3016 as positive.

Classification Report: The evaluation metrics for model performance are precision, recall and F1-score.

Model	Classification	Precision	Reall	F1-score
Fig.3	0 (Negative)	0.89	0.93	0.93
	1 (Positive)	0.93	0.88	0.93
Fig.4	0 (Negative)	0.96	0.13	.0.23
	1(Positive)	0.78	1.00	0.88

Table 2.

- **Precision**: shows the proportion of predicted reviews that were positive from the above classification report fig.3 and fig4 shows that the model performs well in classifying both sentiment reviews with almost the same accuracy.
- **Recall**: Fig.3, the recall for 0 (negatives) is 0.93 while for 1 (positive) 0.88 this indicates that the model performs well in classifying both sentiment reviews with almost the same accuracy. Fig. 4., 0.13 recall means that out of every 100 positives reviews in test data the model only correctly identified 13 while 1.00 means all positive sample are correctly classified.
- **F1-score:** Fig.3. 0.93 for both classes of reviews depicting a good balance between precision and recall. Fig 4. 0.23 means the model could not classify the classes correctly and 0.88 model performs well.

.

Logistic Regression Evaluation

```
Classification Report (Logistic Regression):
              precision    recall  f1-score   support

           0       0.90      0.96      0.93      3021
           1       0.96      0.90      0.93      3071

    accuracy                           0.93      6092
   macro avg       0.93      0.93      0.93      6092
weighted avg       0.93      0.93      0.93      6092
```

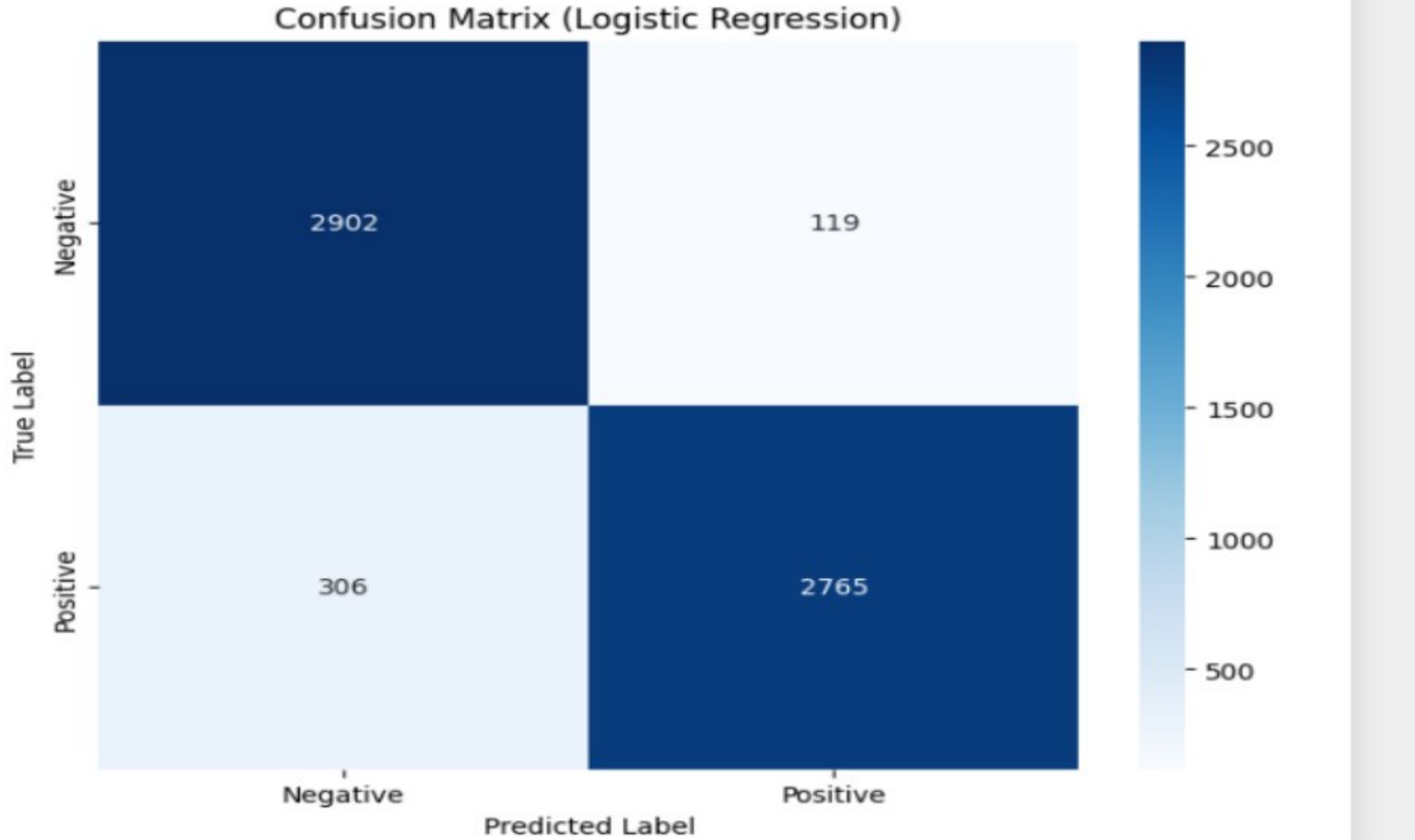

Fig.6. Classification Report and Confusion Matrix for Logistic Regression Classifier.

```
Classification Report (Logistic Regression with TF-IDF):
              precision    recall  f1-score   support

           0       0.88      0.70      0.78       978
           1       0.91      0.97      0.94      3022

    accuracy                           0.90      4000
   macro avg       0.89      0.83      0.86      4000
weighted avg       0.90      0.90      0.90      4000

Confusion Matrix (Logistic Regression with TF-IDF):
[[ 681  297]
 [  92 2930]]
```

Fig.7. Classification Report (Logistic Regression with TF-IDF)

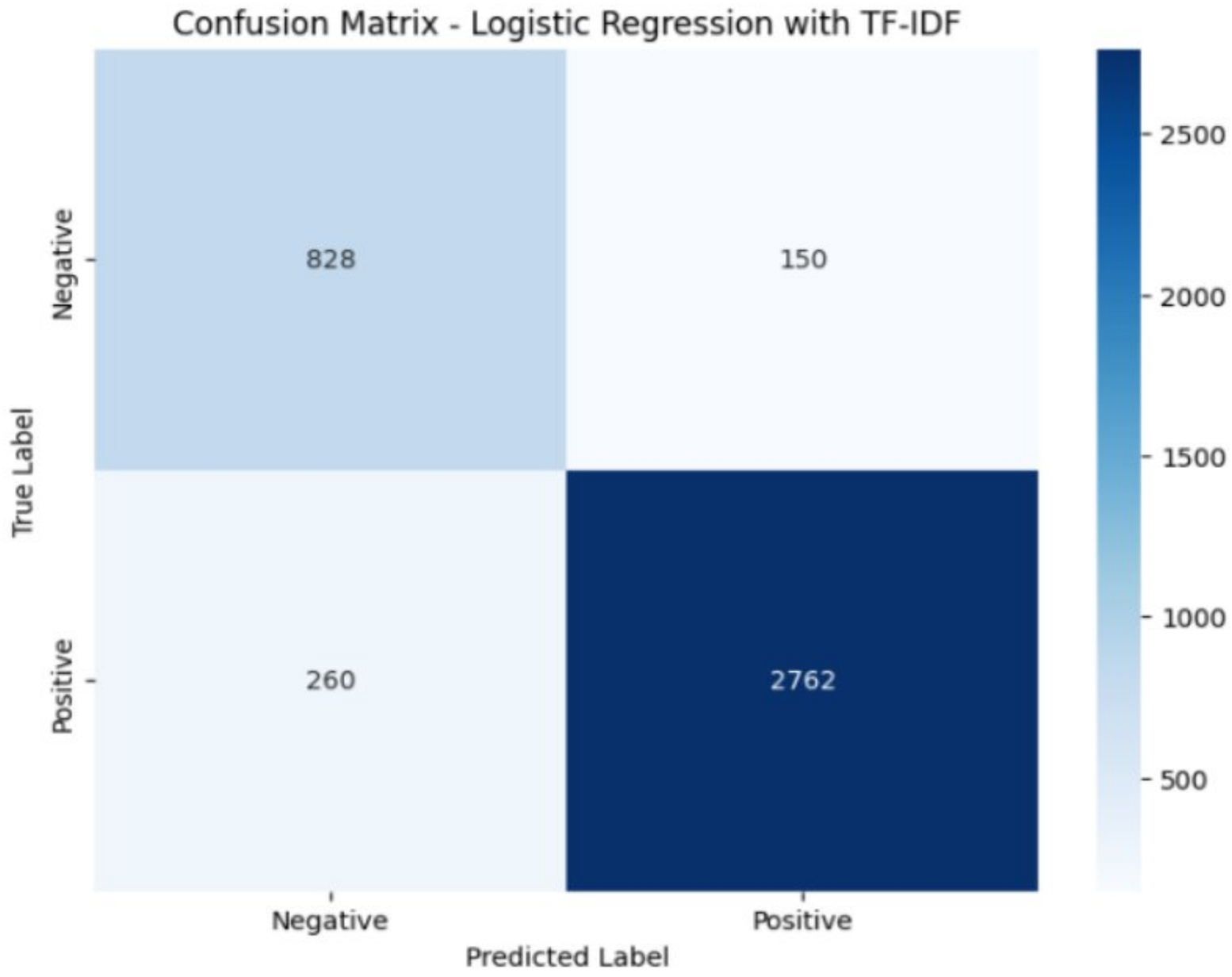

Fig.8. Confusion Matrix (Logistic Regression with TF-IDF)

Model	Classification	Precision	Reall	F1-score
Fig.6	0 (Negative)	0.90	0.96	0.93
	1 (Positive)	0.96	0.90	0.93
Fig.7	0 (Negative)	0.88	0.70	.0.78
	1(Positive)	0.91	0.97	0.94

Table 3. Logistic Regression

- **Precision fig.6 and 7** show the proportion of predicted reviews that were positive from the above classification report it shows that the model performs well in classifying both sentiment classes with almost the same accuracy.
- **F1-score:** Fig.6. 0.93 for both classes of reviews depicting a good balance between precision and recall. Fig 7. 0.78 and 0.94 means the model could classify the classes correctly by 78% and 94% which makes the model performs well.

RNN Model Evaluation

- **Precision fig.9** show the proportion of predicted reviews that were positive from the above classification report it shows that the model performs well in classifying both sentiment classes with almost the same accuracy.
- **F1-score:** Fig 9. 0.73 and 0.89 means the model could classify the classes correctly by 73% and 89% which makes the model perform well.

Model	Classification	Precision	Reall	F1-score
Fig.9	0 (Negative)	0.63	0.86	0.73
	1 (Positive)	0.95	0.84	0.89

Table 4.

Classification Report:

	precision	recall	f1-score	support
0	0.63	0.86	0.73	978
1	0.95	0.84	0.89	3022
accuracy			0.84	4000
macro avg	0.79	0.85	0.81	4000
weighted avg	0.87	0.84	0.85	4000

Fig.9. Classification Report of RNN Classifier

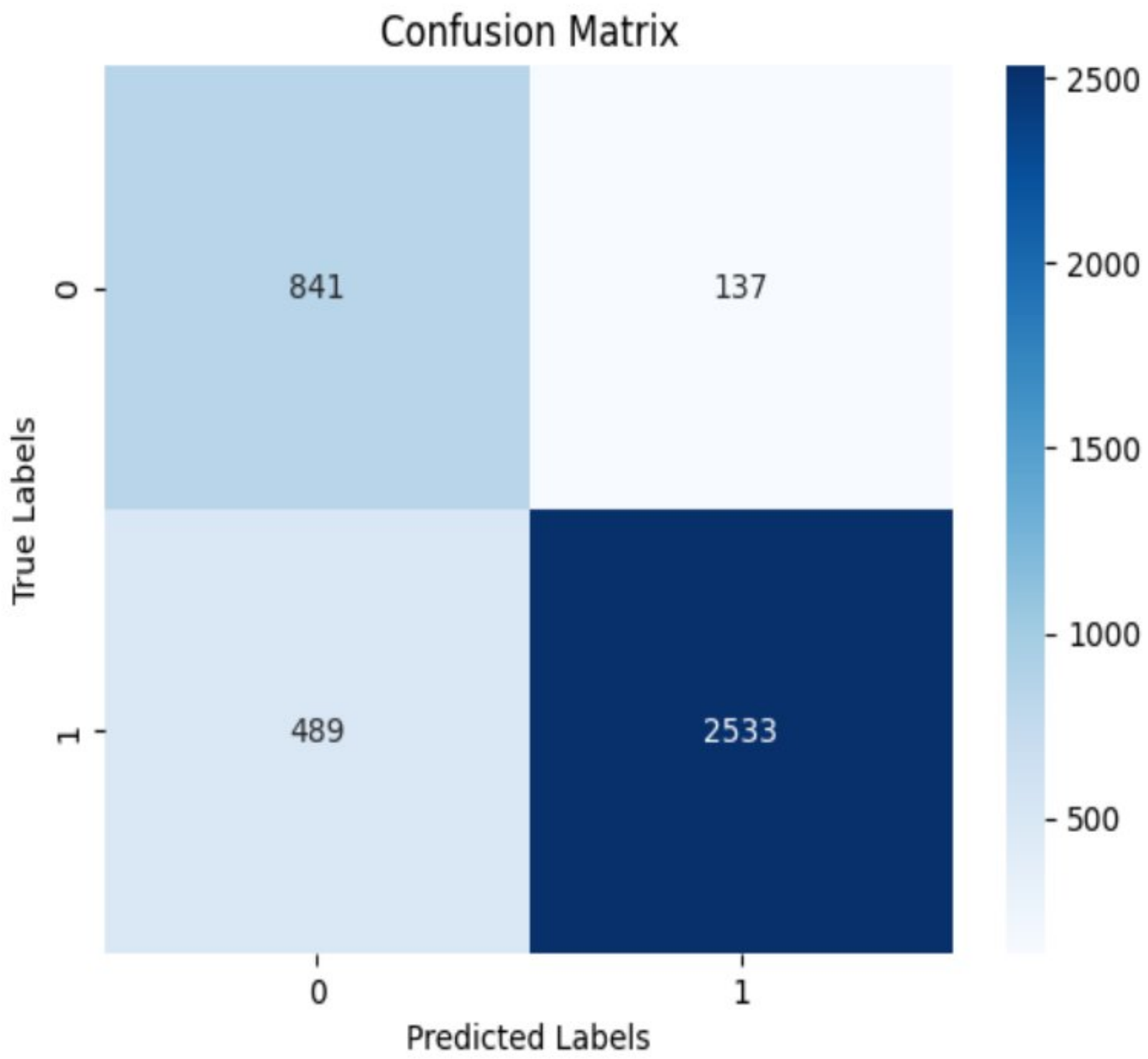

Fig.10. Confusion Matrix of RNN Classifier

From fig. 11 below The Test loss is 0.5563 means the model did not perform well while the Test accuracy of 0.7555 shows that the model is able to predict correctly by 75.55**%**

	precision	recall	f1-score	support
0	0.00	0.00	0.00	978
1	0.76	1.00	0.86	3022
accuracy			0.76	4000
macro avg	0.38	0.50	0.43	4000
weighted avg	0.57	0.76	0.65	4000

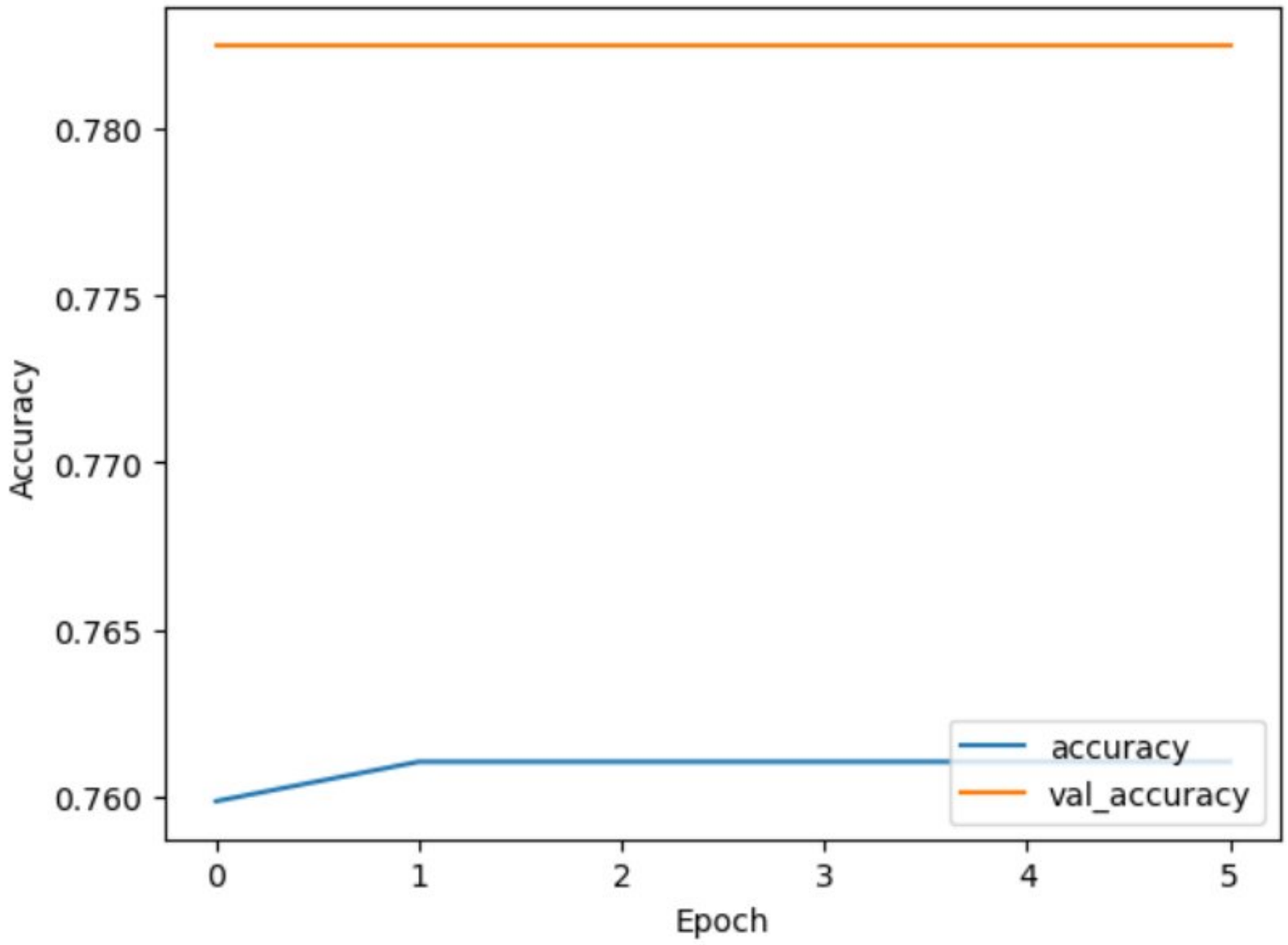

Fig.11. Classification Report and Loss and Accuracy of RNN TF-IDF

Long Short – Term Memory (LSTM)

Precision fig.12 and 14 show the proportion of predicted reviews that were positive from the above classification report it shows that the model performs well in classifying both sentiment classes with almost the same accuracy.

F1-score: Fig 12 and 14. means the model could classify the classes correctly by 80% ,93%, 79% and 94% respectively which makes the model perform well.

Model	Classification	Precision	Reall	F1-score
Fig.12	0 (Negative)	0.76	0.85	0.80
	1 (Positive)	0.95	0.91	0.93
Fig.14	0 (Negative)	0.82	0.77	.0.79
	1(Positive)	0.93	0.94	0.94

Table 5

Classification Report:

	precision	recall	f1-score	support
0	0.76	0.85	0.80	978
1	0.95	0.91	0.93	3022
accuracy			0.90	4000
macro avg	0.85	0.88	0.87	4000
weighted avg	0.90	0.90	0.90	4000

Fig.12. Classification Report of LSTM Classifier

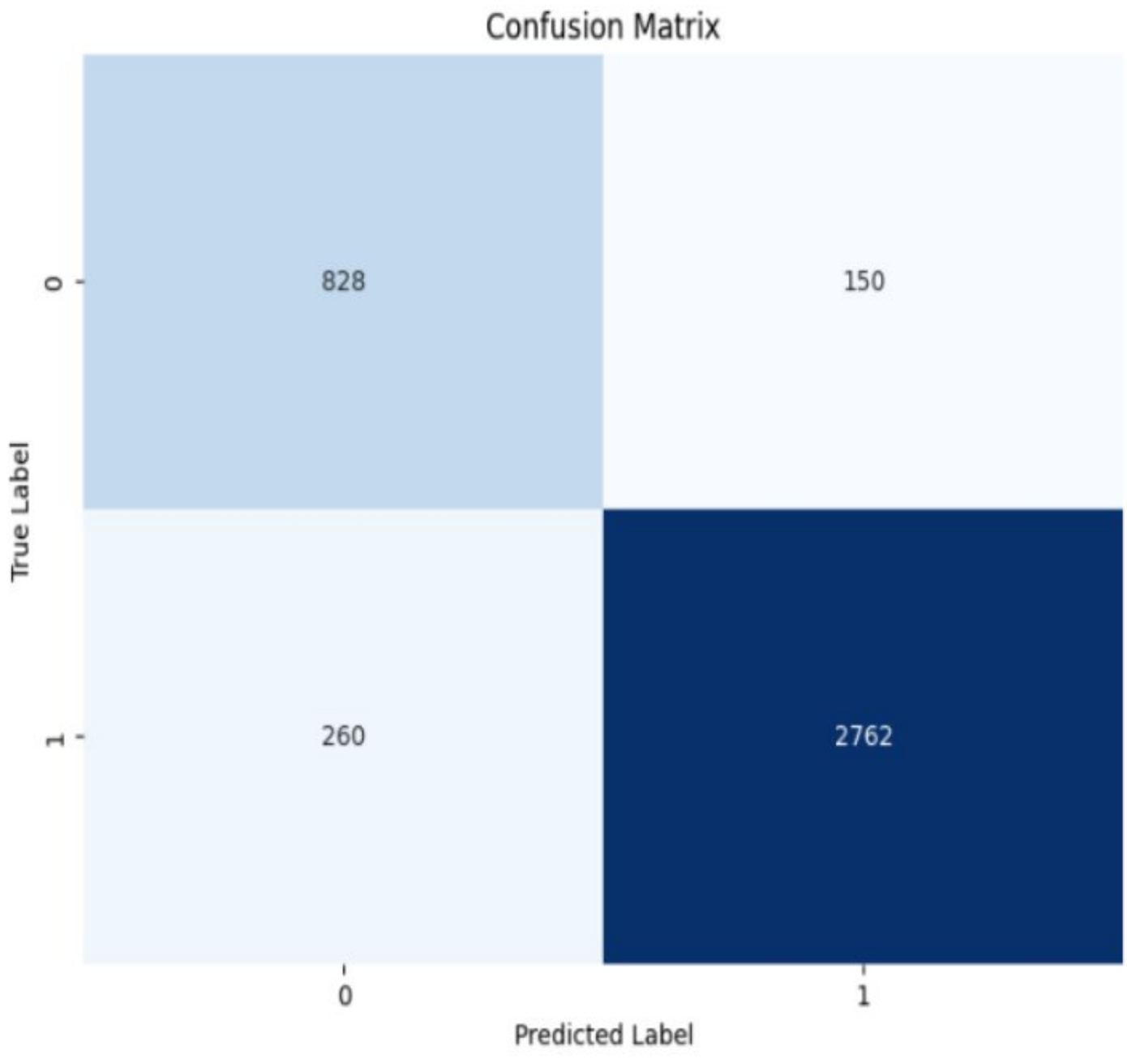

Fig.13. Confusion Matrix of LSTM Classifier

```
Classification Report (Dense NN with TF-IDF):
              precision    recall  f1-score   support

           0       0.82      0.77      0.79       978
           1       0.93      0.94      0.94      3022

    accuracy                           0.90      4000
   macro avg       0.87      0.86      0.86      4000
weighted avg       0.90      0.90      0.90      4000
```

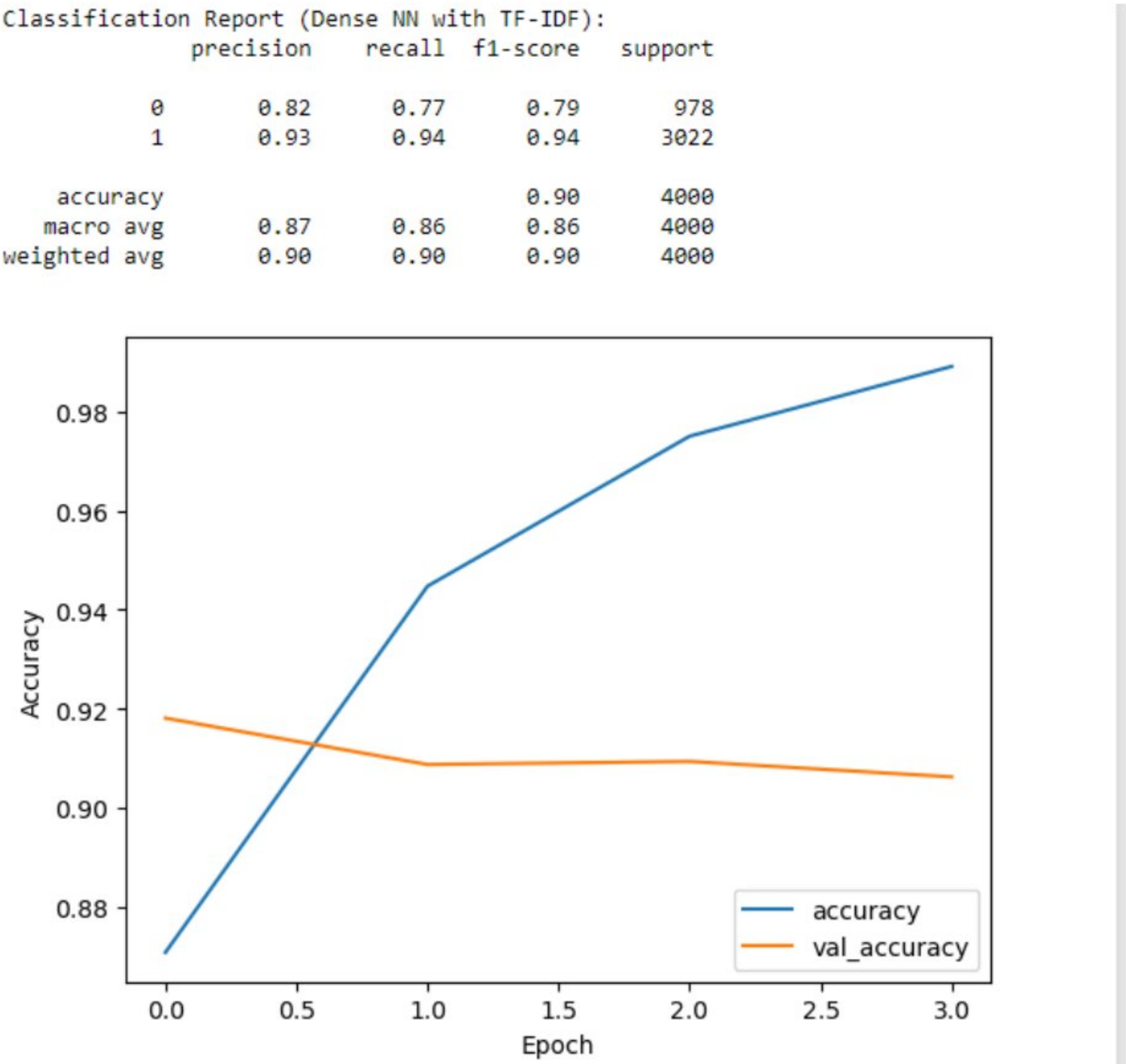

Fig.14. Classification Report and Accuracy of LSTM TF-IDF

REFERENCE

Liddy, E.D., 2001. Natural language processing.

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

Ruales, J., 2011. Recurrent neural networks for sentiment analysis. *IEEE. Colombia: Colombia University*.Ruale

Mikolov, T., Karafiát, M., Burget, L., Cernocký, J. and Khudanpur, S., 2010, September. Recurrent neural network based language model. In *Interspeech* (Vol. 2, No. 3, pp. 1045-1048).

Wang, X., Liu, Y., Sun, C.J., Wang, B. and Wang, X., 2015, July. Predicting polarities of tweets by composing word embeddings with long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th*

International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (pp. 1343-1353).

Liao, S., Wang, J., Yu, R., Sato, K. and Cheng, Z., 2017. CNN for situations understanding based on sentiment analysis of twitter data. *Procedia computer science*, *111*, pp.376-381.