**CHAPTER FOUR**

# IMPLEMENTATION, RESULT AND DISCUSSION

## 4.1 Introduction

This chapter presents the result of the implementation performed in this study and its evaluation metrics employed to validate the result and performance of the model. It also provides a detailed discussion of the results and findings of the model.

## 4.2 Model Implementation

The model does a job of classifying the sentiment into two types:

1. Positive Review.
2. Negative Review.

The classification is done using the algorithm as depicted in {add the model architecture here: figure 3.11}. this process includes the step we took such as text cleaning, tokenization, and model training, leading to the creation of a predictive model that can classify sentiments in text data. In this study, the implementation of the model was carried out using the python environment with its required frameworks, such as; Pandas, NumPy, Seaborn, Matplotlib, Sklearn, keras, tensorflow.

### 4.2.1 Jupyter Notebook

Jupyter Notebook (previously IPython Notebooks) is an online intelligent computational environment for creating notebook documents. The "notebook" term can informally refer to various elements, for the most part, the Jupyter web application, Jupyter Python web server, or Jupyter document relying upon setting. Users may create and share documents with live code, equations, visuals, and narrative text with the open-source Jupyter Notebook program. A Jupyter Notebook archive is a JSON record, containing an ordered list of input/output cells

which can contain code, text (using Markdown), mathematics, plots, and rich media and usually ends with the ".ipynb" extension. It can be used for data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning and much more. The Notebook has support for over 40 programming languages, including Python, R, Julia, Scalia and many more. Notebooks can be shared with others using email, Dropbox, GitHub and the Jupyter Notebook Viewer. It also offers big data tools for Apache Spark from Python, R and Scala using tools such as Pandas, Scikit-learn, ggplot2, TensorFlow and so on. Figure 4.1 shows Jupyter Notebook Environment.

### 4.2.2 VS Code

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes (such as C++, C#, Java, Python, PHP, Go, .NET). Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Figure 4.1 shows the Visual Studio Code Environment.

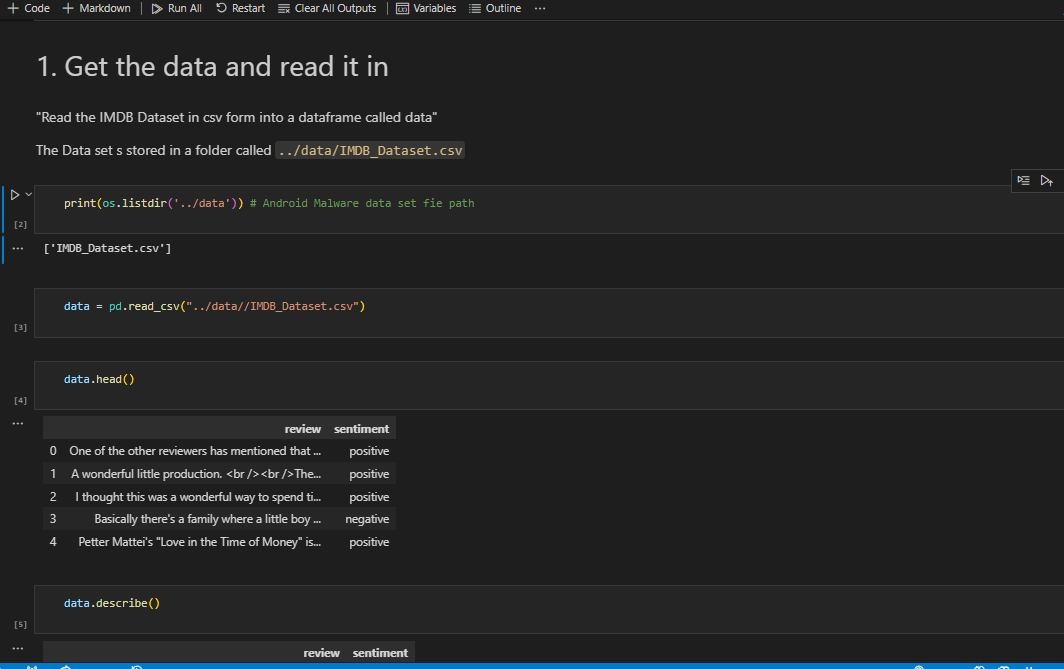


Figure 4.1: Jupyter notebook embedded in a VS Code Environment

## 4.3 Model Evaluation

Model evaluation assesses the performance of a machine learning model by comparing its predictions to the actual outcomes. Common evaluation metrics for sentiment classification models include accuracy, precision, recall, F1 score, running time and area under the ROC curve (AUC-ROC). One can acquire a better understanding of the categorization model's successes and failures by calculating a confusion matrix. Two by Two confusion matrix is employed in this research work because the number of correct and incorrect predictions are summarized with count values and broken down by each class. Mathematical equations are then applied to the number of correct and incorrect count values to evaluate the performance percentage. Four fundamental properties (numbers) make up the confusion matrix, which is used to provide the classifier's measurement parameters. These four numbers are: True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN)

1. True Positives (TP) − It is the case when both the actual class and the predicted class of data point is 1.
2. True Negatives (TN) − It is the case when both the actual class and the predicted class of the data point are 0.
3. False Positives (FP) − It is the case when the actual class of data point is 0 and the predicted class of data point is 1.
4. False Negatives (FN) − It is the case when the actual class of data point is 1 and the predicted class of data point is 0.

Also, Performance evaluation is an important aspect of the machine learning process. However, it is a complex task. It, therefore, needs to be conducted carefully for the application of machine learning to be reliable. Some of the metrics utilized in the execution of this project include the following;

1. **Accuracy**

It is the most common performance metric for classification algorithms. It may be defined as the number of correct predictions made as a ratio of all predictions made. It can be easily calculated by a confusion matrix with the help of;

Accuracy = 𝑇𝑃+𝑇𝑁

𝑇𝑃+𝐹𝑃+𝐹𝑁+𝑇𝑁

4.1

1. **Precision**

It is used in document retrievals and may be defined as the number of correct documents returned by our ML model. It can be easily calculated by a confusion matrix with the help of;

Precision = 𝑇𝑃

𝑇𝑃+𝐹𝑃

1. **Recall or Sensitivity**

4.2

Recall may be defined as the number of positives returned by our ML model. It can be easily calculated by a confusion matrix with the help of;

Recall = 𝑇𝑃

𝑇𝑃+𝐹𝑁

4.3

1. **F1 Score**

This score will give us the harmonic mean of precision and recall. Mathematically, the F1 score is the weighted average of precision and recall. The best value of F1 would be 1 and the worst would be 0. It can be easily calculated by a confusion matrix with the help of;

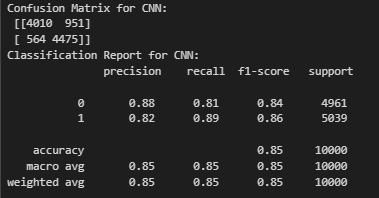
F1 = 2 ∗ (precision ∗ recall) / (precision + recall) 4.4

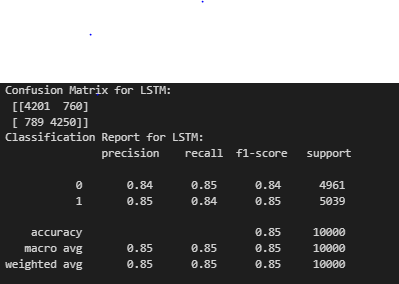
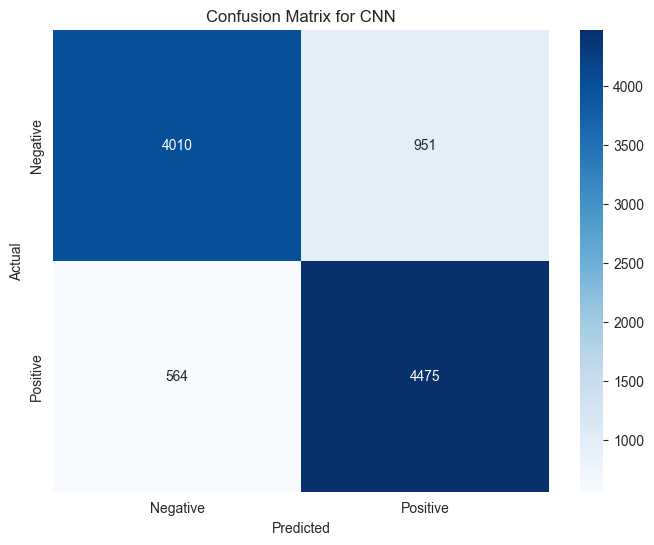
F1 score is having an equal relative contribution of precision and recall. **v Area Under ROC Curve**

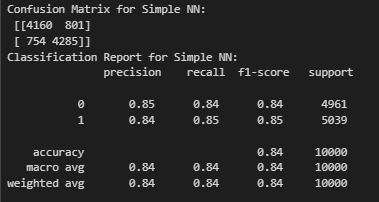
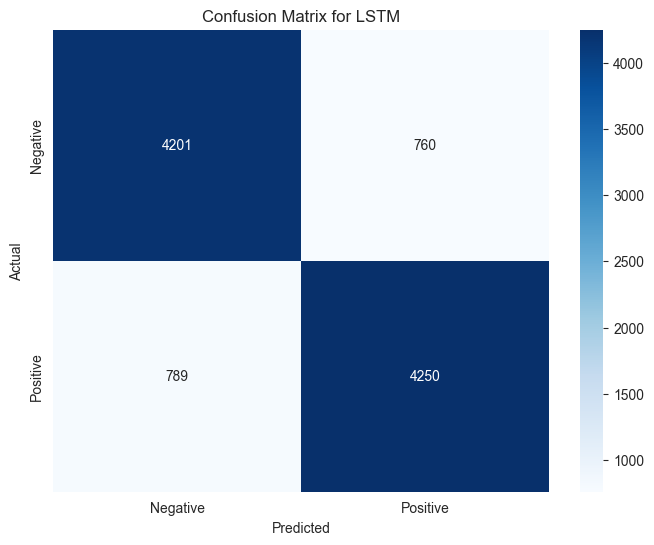
AUC (Area Under Curve)-ROC (Receiver Operating Characteristic) is a performance metric, based on varying threshold values, for classification problems. As the name suggests, ROC is a probability curve and AUC measures the reparability. In simple words, the AUC-ROC metric will tell us about the capability of the model in distinguishing the classes. Higher the AUC, the better the model. Mathematically, it can be created by plotting TPR (True Positive Rate) i.e., Sensitivity or recall vs FPR (False Positive Rate) i.e., 1-Specificity, at various threshold value.

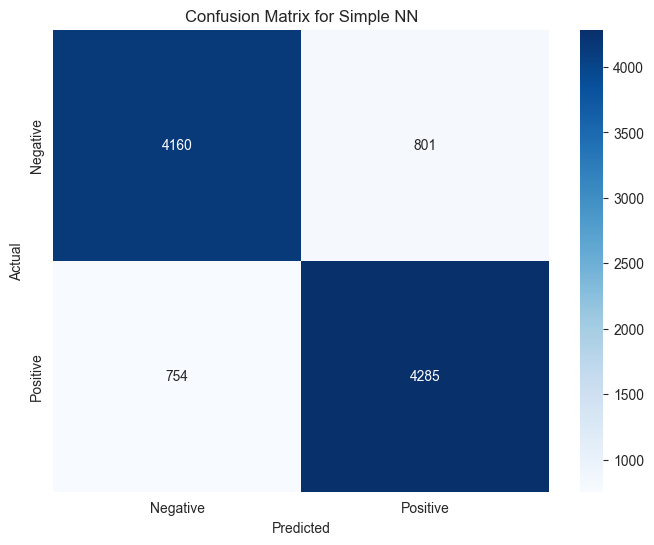
## 4,4 Result and Discussion

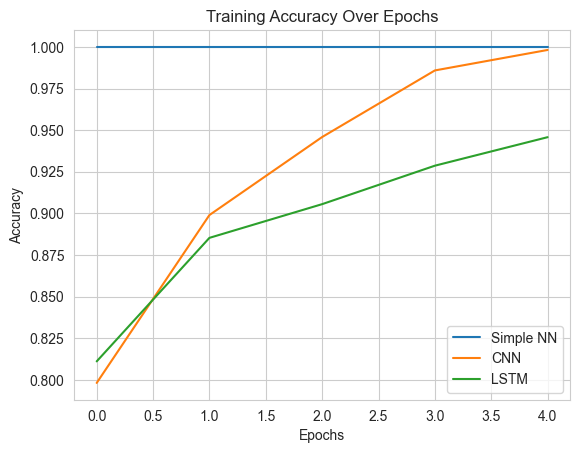
We employed three distinct neural network architectures for sentiment classification on online product reviews. Simple Neural Network (Simple NN), Convolutional Neural Network (CNN), and Long Short-Term Memory Network (LSTM). The evaluation metrics for each model, as depicted in Table 4.1 provide a comprehensive overview of their performance. The Simple NN achieved an accuracy of 84.18%, precision of 84.40%, recall of 84.16%, and an F1-Score of 84.28%. The CNN demonstrated competitive results with an accuracy of 84.85%, precision of 82.47%, recall of 88.81%, and an F1-Score of 85.52%. Meanwhile, the LSTM model exhibited an accuracy of 84.51%, precision of 84.83%, recall of 84.34%, and an F1-Score of 84.59%. Comparing these models, the CNN displayed the highest recall at 88.81%, making it particularly adept at correctly identifying positive sentiments. On the other hand, the Simple NN and LSTM exhibited balanced performances across precision, recall, and F1-Score. The AUROC graph in Figure 4.8 shows the trade-off between the true positive rate (TPR) and false positive rate (FPR) at various threshold settings for each classifier AUC-ROC values further support the models' discriminatory power, with CNN leading at 92.75%, followed closely by Simple NN (92.30%) and LSTM (92.02%). These results underscore the models' ability to distinguish between positive and negative sentiments. The computational efficiency, measured in running time, varied among the models. Simple NN demonstrated the longest runtime at 88.86 seconds, while CNN and LSTM exhibited significantly shorter durations at 12.11 and 20.27 seconds, respectively. Figure 4.9 is a heat map illustrating the performance analysis of the individual classifier in comparison with other classifiers











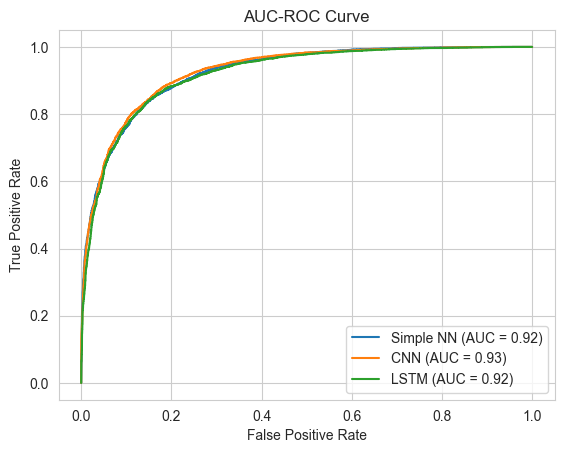


Figure 4.8

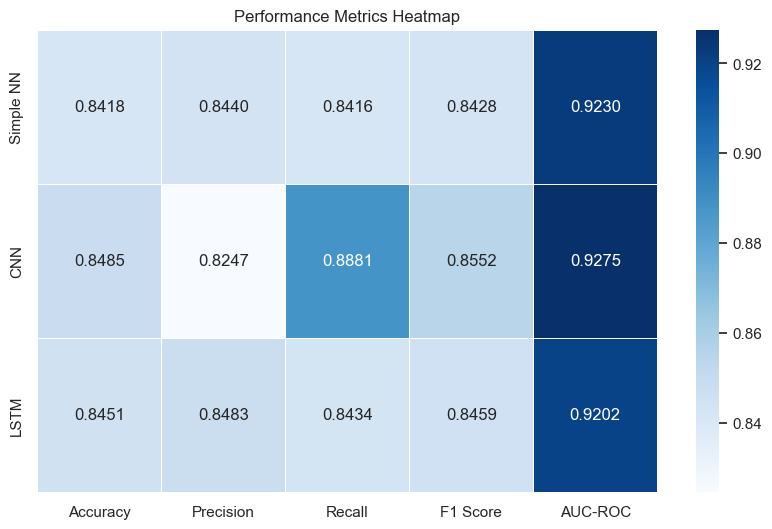


Figure 4.9

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Accuracy | Precision | Recall | F1 Score | AUC-ROC | Running Time (s) |
| SNN | 0.8418 | 0.843980 | 0.841635 | 0.842806 | 0.922953 | 88.859657 |
| CNN | 0.8485 | 0.824733 | 0.888073 | 0.855232 | 0.927475 | 12.114410 |
| LSTM | 0.8451 | 0.824733 | 0.843421 | 0.845855 | 0.920222 | 20.268860 |

Table 4.1: overview of the model performance.