**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 and Scalia. 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.

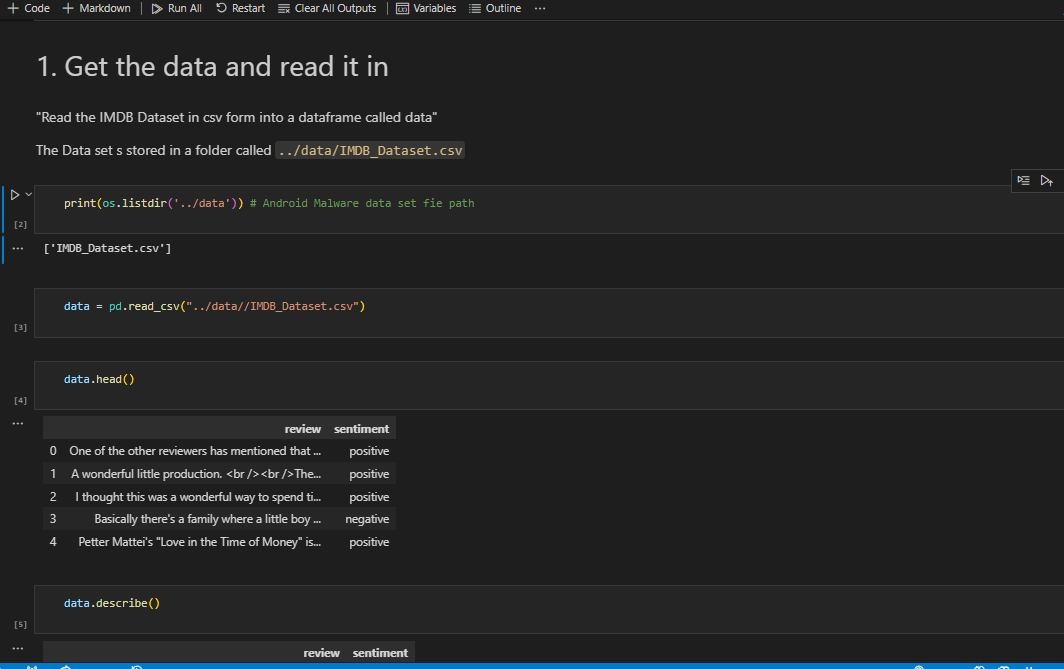


Figure 4.1: Jupyter notebook embedded in a VS Code Environment

## 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, 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 = 𝑇𝑃6𝑇𝑁

𝑇𝑃6𝐹𝑃6𝐹𝑁6𝑇𝑁

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

𝑇𝑃6𝐹𝑃

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

𝑇𝑃6𝐹𝑁

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.

# Result and Discussion

From the experiment carried out for the aforementioned techniques, Figures 4.3, 4.4, 4.5 and

4.6 show the confusion matrix result of the Support Vector Machine, K-Nearest Neighbor, Random Forest and Ensemble respectively, in which the ensemble proved best among them. The ensemble (SVM, KNN, RF) had an Accuracy of 97.41%, Precision of 97.73%, Recall of 95.31% and F1-Score of 96.51. Support Vector Machine had an Accuracy of 70.93%, Precision of 56.48%, Recall of 98.67% and F1-Score of 71.84%. K-Nearest Neighbor had an Accuracy of 76.82%, Precision of 65.50%, Recall of 80.97% and F1-Score of 72.42%. Random Forest had an Accuracy of 79.18%, Precision of 67.26%, Recall of 86.90% and F1-Score of 75.83%. In Table 4.1, the performance of the various machine learning models was compared and evaluated. K Nearest Neighbors had the lowest computations time of 31.29 secs while Support Vector Machines had the highest with 48.37 secs. The ensemble had the most Accuracy with 97.41% while Support Vector Machines had the lowest with 70.93%. The information represented in Table 4.1 was further presented for graphical visualization as shown in Figure

4.7. 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, with the ensemble showing a higher value prospect of 0.993, indicating a better model. Figure 4.9 is a heat map illustrating the performance analysis of the individual classifier in comparison with other classifiers