Name : Kamrul Golder

Batch : B7

Requirements for Sentiment Analysis Model and Console App

1. Software Requirements:

- **Python**: The programming language used to implement the model and the console app.
- **Google Colab**: An online environment for running Python code, especially for deep learning models.
- **Libraries**:
  - **TensorFlow/Keras**: For building and training the deep learning model.
  - **scikit-learn**: For preprocessing data, encoding labels, splitting data, and evaluation metrics.
  - **Pandas**: For handling data, especially the dataset of user comments.
  - **NumPy**: For numerical operations, such as array handling.
  - **Matplotlib/Seaborn**: For plotting visualizations like confusion matrices and ROC curves.
  - **OpenCV (optional)**: If working with image-based datasets (not used in this code).

2. Data Requirements:

- **Labeled Dataset**: A dataset with labeled user comments, where each comment is associated with a sentiment label (positive, neutral, or negative). This can be in formats such as CSV or Excel.
  - Example dataset columns: Comment, Status (label: positive, neutral, negative).
- **Data Preprocessing**:
  - Cleaning: Removing missing values, ensuring all comments are strings.
  - Tokenization: Converting text into sequences of numbers.
  - Padding: Making all input sequences of equal length.

3. Hardware Requirements:

- **CPU/GPU**: While a CPU can be used for training, using a GPU can speed up the model training process, especially for larger datasets. Google Colab provides free access to GPUs.
- **RAM**: Sufficient RAM (minimum 4GB) is needed for processing large datasets and training models.

4. Model Requirements:

- **Text Classification Model**: The model should be capable of handling text data and classifying it into multiple categories (positive, neutral, negative). This includes:
    - **Embedding Layer**: To convert words into dense vector representations.
    - **Dense Layers**: For classification.
    - **Activation Functions**: relu for hidden layers and softmax for output layer (multi-class classification).
- **Training Data**: Sufficient labeled data for training the model, split into training, validation, and test sets.
- **Evaluation Metrics**: Accuracy, confusion matrix, classification report, and ROC curve for performance evaluation.

5. Console App Requirements:

- **Input**: The app should allow users to input text for sentiment analysis.
- **Output**: The app should predict whether the input text is positive, neutral, or negative.
- **Exit Functionality**: Users can type "exit" to stop the app.
- **User-Friendly Interface**: While console-based, the app should provide clear instructions and responses, making it easy to interact with.

6. Performance Considerations:

- The model should achieve reasonable accuracy on the test set (e.g., above 80% for a typical sentiment analysis task).
- The app should provide real-time predictions with minimal latency.

7. Deployment Considerations:

- **Model Persistence**: Once the model is trained, it should be saved and loaded for future use without retraining.
- **Scalability**: While the console app is suitable for small-scale usage, the model can be expanded for larger-scale applications such as integration with web or mobile platforms.

By meeting these requirements, the sentiment analysis model and its console app can offer an efficient, real-time solution for text classification tasks.