Name:Sai varun konda

Email:saivarunkonda007@gmail.com

1. Data Loading and Preprocessing:
   * Loads a CSV file containing text and sentiment labels.
   * Cleans text data by removing HTML tags, punctuation, numbers, and stop words.
   * Tokenizes text into individual words.
   * Lemmatizes words to their base forms.
   * Converts text to numerical sequences using a tokenizer.
   * Pads sequences to a uniform length for model compatibility.
2. Model and Training:
   * Constructs a neural network architecture with an embedding layer, an LSTM or RNN layer, and a dense output layer for sentiment classification.
   * Compiles the model with appropriate loss function and optimizer.
   * Trains the model on the preprocessed text and sentiment labels.
3. Evaluation:
   * Splits data into training and testing sets.
   * Evaluates model performance on both sets using accuracy metrics.
   * Generates a confusion matrix and classification report for detailed analysis.
4. Prediction:
   * Defines a preprocess\_and\_predict() function to handle new text input:
     + Preprocesses text using the same steps as in training.
     + Loads the trained model from a pickle file.
     + Converts preprocessed text to vectorized sequences.
     + Reshapes sequences to match model input requirements.
     + Makes a sentiment prediction using the model.
     + Prints the preprocessed text and predicted sentiment.
5. Interactive Usage:
   * Prompts the user to enter text or type "exit" to quit.
   * Preprocesses and predicts sentiment for each input until "exit" is entered.

Key Libraries and Functions:

* Pandas: Reads CSV data, handles dataframes.
* NumPy: Numerical computations and array operations.
* nltk: Natural language processing tasks (tokenization, lemmatization, stop words).
* BeautifulSoup: Extracts text from HTML content.
* Keras: Deep learning framework for model creation and training.
* TensorFlow: Backend for Keras, provides tensor operations.
* Scikit-learn: Model evaluation metrics (confusion matrix, classification report)

**Model training takes two hours you can directly run last second cell and run the chatbot.**

I tried with vectorizer and embedding I got accuracy of 0.5 to 0.6 I used keras tunner to give the best output by varying the hyperparameters then I used the embedding layer in model and got 0.9955 accuracy.