Project Title: Text Sentiment Analysis using Deep Learning (Twitter Dataset)

Objective:

To classify text (e.g., tweets) into different sentiment categories (e.g., positive, negative, neutral) using deep learning with a neural network.

Key Libraries Used:

- TensorFlow / Keras - Deep learning model building
- Pandas - Data handling
- NumPy - Numerical operations
- Matplotlib, Seaborn - Visualization
- scikit-learn - Data splitting, preprocessing, evaluation
- Openpyxl - Excel file reading
- Pickle - Saving/loading model components

Dataset Details:

- Loaded from Excel: twitter_training.xlsx
- Columns used: text, Label
- Multiclass classification (labels like Positive, Negative, Neutral, etc.)

Preprocessing Steps:

1. Splitting: 70% training, 15% validation, 15% testing
2. Label Encoding: LabelEncoder used to convert text labels to numeric
3. Text Tokenization: Tokenizer converts text into integer sequences, padded to max_length = 100

Model Architecture:

Sequential Model:

- Embedding Layer: Input Dim: vocab_size, Output Dim: 16, Input Length: 100

- GlobalAveragePooling1D: Averages embeddings across time dimension

- Dense Layer: 32 neurons, ReLU activation

- Output Layer: Softmax activation (num_classes units)

Model Summary:

- Embedding layer: learns word vector representations

- Global pooling: reduces dimensionality

- Dense + softmax: classifies sentiment

- Optimizer: adam

- Loss: categorical_crossentropy

- Epochs: 19

Training & Evaluation:

- Model trained on 70% data for 19 epochs

- Evaluated using: Accuracy, Confusion Matrix, Classification Report (precision, recall, F1-score)

- Accuracy and loss plotted across epochs

Testing Phase:

- Model evaluated on test set using accuracy_score, confusion_matrix, classification_report

- Example sentences tested for real-world performance

Model Persistence:

- text_sentiment_model.h5 - Full trained model

- tokenizer.pkl - Tokenizer object

- label_encoder.pkl - Encoded label mapping

- label_classes.npy - Optional class names

Real-Time Prediction:

- Accepts user input in real time
- Text preprocessed -> tokenized -> padded -> predicted
- Outputs predicted label + confidence score

Key Points for Viva:

- Tokenizer: Converts text to integers; trained on training data
- Embedding Layer: Learns dense vector representations for words
- Pooling: GlobalAveragePooling1D reduces sequence to fixed-size
- Loss Function: categorical_crossentropy (multi-class)
- Optimizer: Adam (adaptive learning rate)
- Activation Functions: ReLU (hidden), Softmax (output)
- Final Metrics: Accuracy, precision, recall, F1-score
- Epochs: 19 (validation curve to monitor overfitting)
- Use Case: Twitter monitoring, feedback analysis, etc.
- Deployment Readiness: Model saved and loadable for real-time prediction

1-Minute Viva Answer:

"I built a deep learning model using Keras to classify text sentiments from a Twitter dataset. After preprocessing and splitting the data, I used a Sequential model with an Embedding layer, GlobalAveragePooling1D, and Dense layers. The model was trained for 19 epochs using categorical crossentropy loss and Adam optimizer. I evaluated it using accuracy, confusion matrix, and classification report. Finally, I saved the model, tokenizer, and label encoder for real-time prediction, and implemented a live input feature for predicting sentiment with confidence."