Technical Documentation for Sentiment Analysis Program

Introduction

This technical documentation provides an overview of a Python program for sentiment analysis. The program reads text data from a CSV file, performs preprocessing on the text, applies sentiment analysis using logistic regression, and allows users to input sentences for sentiment analysis. The sentiment analysis is conducted using the TextBlob library.

Libraries and Dependencies

The program relies on several Python libraries and dependencies. Ensure these are installed in your Python environment:

- `pandas`: For data manipulation and analysis.
- `numpy`: For numerical computations.
- `re`: For regular expressions and text cleaning.
- `seaborn` and `matplotlib`: For data visualization.
- `textblob`: For sentiment analysis.
- `nltk`: For natural language processing, including tokenization, stemming, and lemmatization.
- `wordcloud`: For creating word clouds.
- `sklearn`: For machine learning and logistic regression.
- `nlpaug`: For text augmentation.
- `joblib`: For model serialization.

Program Flow

1. **Data Loading and Preprocessing**:

    - Import necessary libraries and dependencies.
    - Read a CSV file named 'sentiment_test_cases.csv' using pandas.
    - Perform various preprocessing steps, including:
        - Lowercasing the text.
        - Removing URLs, usernames, and special characters.
        - Tokenization (word splitting).
        - Removing stopwords (common words like "the," "is," etc.).
        - Stemming and lemmatization of words.
    - Augment the text data to increase the dataset size.

2. **Sentiment Analysis**:
    - Use the TextBlob library to analyze sentiment. This library provides a sentiment polarity score.
    - Categorize sentiment as 'Positive,' 'Neutral,' or 'Negative' based on the polarity score.
    - Visualize the distribution of sentiment labels using count plots and pie charts.
    - Display the most frequent words in positive sentiment using a word cloud.
3. **Feature Extraction**:
    - Utilize the CountVectorizer to convert text data into a numerical format suitable for machine learning.
    - Explore the number of features and example features.
4. **Model Building and Training**:
    - Split the dataset into training and testing sets.
    - Create and train a logistic regression model for sentiment analysis.
    - Evaluate model performance using accuracy, confusion matrix, and classification report.
    - Visualize the confusion matrix using a confusion matrix display.
5. **Model Serialization**:
    - Serialize the trained logistic regression model using joblib.
    - Save the model to a file named 'logistic_regression_sentiment_model.pkl.'
    - Load the model from the file to perform sentiment analysis on user input.
6. **User Input Analysis**:
    - Accept user input in the form of a sentence.
    - Preprocess the user input in the same way as the training data.
    - Analyze the sentiment of the user input using the loaded model.
    - Calculate a confidence score based on the sentiment polarity.
    - Output the sentiment label and confidence score.
7. **Test Data Sentiment Analysis**:
    - Read another CSV file named 'test.csv' containing test data.
    - Preprocess and analyze the sentiment of test data using the loaded model.
    - Create columns for sentiment and confidence scores.
    - Output the results in a DataFrame.

Usage

To use this program, ensure that you have the required libraries installed and that the CSV files ('sentiment_test_cases.csv' and 'test.csv') are in the specified locations. The program allows you to perform sentiment analysis on user input sentences and analyze the sentiment of a test dataset.