README - Sentiment Analysis and Prediction of Tweets

Objective

This script performs sentiment analysis on tweet data using machine learning. It includes:

1. Preprocessing the tweet text and labeling sentiments.
2. Training a machine learning model to predict tweet sentiments.
3. Visualizing the sentiment distribution and model performance.

Prerequisites

1. **Python** installed on your system.
2. Install required libraries using pip:
3. pip install pandas textblob matplotlib seaborn scikit-learn
4. **Dataset**:
   - A CSV file (tweets_with_sentiment.csv) containing a text column with tweets.
   - The script will automatically generate a label column (1 for positive, 0 for negative) based on sentiment polarity.

How to Use

1. **Prepare the Data**: Ensure that your CSV file (tweets_with_sentiment.csv) contains a text column with tweet content.
2. **Save the Script**: Save the Python script in a file, e.g., sentiment_analysis.py.
3. **Run the Script**:
4. python sentiment_analysis.py

Script Workflow

1. **Preprocess Data**:
   - The script loads the CSV file.
   - If the sentiment column is not available, it calculates sentiment using TextBlob:
     - **Positive sentiment**: Label 1
     - **Negative sentiment**: Label 0
2. **Train the Model**:
   - The dataset is split into training and testing sets (80/20).
   - The CountVectorizer converts text into numerical features.
   - A **Logistic Regression model** is trained on the text data.
   - The model is evaluated based on accuracy and classification report.
3. **Visualize Results**:

   - **Confusion Matrix**: A heatmap displays the model's true positives, true negatives, false positives, and false negatives.
   - **Prediction Accuracy Plot**: A scatter plot compares actual and predicted sentiment labels for the test set.
   - **Sentiment Distribution**: A bar plot shows the count of positive and negative sentiments in the dataset.
4. **Make Predictions**:
   - You can test the model by providing new text samples for sentiment prediction.

Expected Output

1. **Console Output**:
   - Summary of sentiment scores and the confusion matrix.
   - Classification report with accuracy, precision, recall, and F1-score.
   - Sentiment prediction for test text samples (e.g., "Python is awesome!" and "I hate coding bugs.").
2. **Plots**:
   - **Sentiment Distribution**: A bar chart showing the count of positive vs. negative sentiments in the dataset.
   - **Confusion Matrix**: A heatmap illustrating the classification performance.
   - **Prediction Accuracy Plot**: A scatter plot comparing actual and predicted sentiment labels.

Customization

- **Change Input File**: Replace "tweets_with_sentiment.csv" with your desired dataset file.
- **Adjust Sentiment Threshold**: Modify how sentiments are labeled based on the TextBlob polarity value.
- **Add New Texts for Prediction**: Modify the test_texts list to test the model with different tweets.

Potential Errors

1. **File Not Found**: Ensure the correct path to the CSV file is specified.
2. **Missing Text Data**: Ensure the CSV file has the text column with tweet content.
3. **Model Overfitting**: If your dataset is small, consider adding more data for better accuracy.

Example Output

1. **Confusion Matrix**:
   - A heatmap with counts for true positives, false positives, true negatives, and false negatives.
2. **Prediction Accuracy**:
   - A scatter plot comparing the actual sentiment labels with the predicted ones.
3. **Sentiment Distribution**:

   - A bar chart showing the number of positive and negative sentiment labels in your dataset.

Note: This script is a simple sentiment analysis tool. You can extend it with more advanced models and additional features like handling neutral sentiment or using deep learning approaches for better accuracy.