Sentiment Analysis of Tweets using Logistic Regression

Problem Statement

The task is to build a machine learning model that can **classify tweets as positive or negative** based on their content. This is a binary classification problem in the field of **Natural Language Processing (NLP)**.

The goal is to:

- Understand the process of cleaning and preprocessing tweets
- Build a logistic regression model from scratch
- Extract word-based features and train the model using labeled tweet data
- Evaluate the model performance and analyze errors
- **Source**: `twitter_samples` corpus from NLTK
- **Size**: 10,000 tweets total
  - 5,000 Positive tweets
  - 5,000 Negative tweets
- **Split**:
  - 80% training set → 8,000 tweets (4K positive, 4K negative)
  - 20% test set → 2,000 tweets (1K positive, 1K negative)

Methodology

Preprocessing:

- Tweets were cleaned using `process_tweet()`:
  - Lowercasing, link removal, punctuation stripping
  - Stopword removal using NLTK
  - Stemming with Porter Stemmer

Feature Engineering:

- For each tweet, a 3-dimensional feature vector was created:
  1. Bias Term (set to 1)
  2. Count of positive-associated words in the tweet
  3. Count of negative-associated words in the tweet

Model:

- **Logistic Regression** implemented from scratch:
  - `sigmoid(z)` for probability
  - gradientDescent(X, y, theta) to optimize weights

Results

Accuracy:

- Test Set Accuracy: ~99.5%
- Based on the confusion matrix and prediction plots, the model performs very well on the clean and simple dataset.

Visualizations:

- **Probability Distribution** shows confident predictions clustered around 0 or 1
- **Confusion Matrix** confirms low false positives/negatives

- **Error Analysis** reveals edge-case tweets with ambiguous sentiment near 0.5 decision boundary

Key Learnings

- Logistic Regression can perform well even with simple feature extraction, especially when the data is clean.
- Preprocessing (stemming, removing stopwords) has a significant impact on model performance.
- Most classification errors occur near the threshold (0.5), showing the model is uncertain in some cases.

Future Work & Scope

Improvements:

- Use **TF-IDF**, **word embeddings**, or **contextual models (like BERT)** for better feature extraction.
- Include **neutral sentiment** for a three-class classification.
- Handle sarcasm, irony, emojis, and multilingual tweets.

Model Extension:

- Train on more diverse datasets from real Twitter streams.
- Integrate with a live tweet fetcher (e.g., Tweepy) for real-time analysis.
- Deploy the model as a web app using Flask/Streamlit.

Conclusion

This project successfully demonstrated how to build and train a simple yet effective **logistic regression model** for sentiment classification on tweets. Despite the simplicity of the features, the model achieved high accuracy and offered interpretable predictions. It forms a strong baseline for more advanced NLP models.