Sentiment Analysis Dashboard

Introduction

Sentiment analysis, also known as opinion mining, is a Natural Language Processing (NLP) technique used to determine the sentiment or emotional tone of text data. It is widely used in customer feedback analysis, social media monitoring, and market research. This project implements a **Sentiment Analysis Dashboard** using Streamlit, allowing users to analyze sentiments from text input or batch process large datasets using multiple sentiment analysis models.

Tools and Technologies Used

- **Python** – Primary programming language.
- **Streamlit** – For building the interactive dashboard.
- **NLTK (Natural Language Toolkit)** – Provides the VADER sentiment analysis tool.
- **TextBlob** – Another NLP library for sentiment polarity analysis.
- **Hugging Face Transformers** – Implements deep learning-based sentiment classification using a BERT-based model.
- **Matplotlib & Seaborn** – Used for visualization.
- **WordCloud** – Generates word clouds from textual data.
- **Pandas & NumPy** – Handles data processing and manipulation.

Methodology

This project follows a structured approach:

1. **User Input Handling**
    - Accepts either single-text input or batch processing via CSV file upload.
    - Provides an interactive sidebar for selecting sentiment analysis models.
2. **Sentiment Analysis Models**
    - Implements three different models for sentiment analysis: VADER, TextBlob, and BERT-based Transformer.
3. **Visualization**
    - Generates graphical representations of sentiment distributions using histograms, pie charts, and word clouds.
4. **Deployment**
    - The dashboard is containerized and deployed to the cloud (e.g., GitHub + Streamlit Cloud).

Models Used

1. VADER (Valence Aware Dictionary and sEntiment Reasoner)

- **Library**: NLTK
- **Approach**: Lexicon-based, rule-based method.

- **How It Works**: Uses a predefined sentiment lexicon with intensity values for words. It calculates sentiment polarity scores based on text composition.
- **Output**: Returns a dictionary with positive, neutral, and negative scores along with a compound score for overall sentiment.
- **Best Suited For**: Short, informal text like social media posts and customer reviews.

2. TextBlob

- **Library**: TextBlob (built on NLTK & Pattern)
- **Approach**: Uses a naive Bayes classifier and lexicon-based sentiment polarity.
- **How It Works**:
  - Analyzes sentence structure and assigns a polarity score (-1 to 1) and subjectivity score (0 to 1).
  - A positive polarity indicates positive sentiment, negative for negative sentiment, and zero for neutral sentiment.
- **Best Suited For**: General NLP tasks and basic sentiment classification.

3. BERT-based Transformer

- **Library**: Hugging Face `transformers`
- **Model Used**: `distilbert/distilbert-base-uncased-finetuned-sst-2-english`
- **Approach**: Deep learning-based, trained on the Stanford Sentiment Treebank (SST-2 dataset).
- **How It Works**:
  - Uses the `pipeline` function from `transformers` to classify text as **positive** or **negative**.
  - Outputs a probability score indicating sentiment strength.
- **Best Suited For**: High-accuracy sentiment analysis tasks, especially for longer texts.

Data Visualization

1. **Sentiment Distribution**
   - Uses Seaborn's histogram (histplot) to show the distribution of sentiments.
   - KDE (Kernel Density Estimation) enables better visual representation.
2. **Word Cloud**
   - Generates a visual representation of frequently occurring words.
   - Uses the `WordCloud` library with Matplotlib for visualization.
3. **Pie Chart**
   - Represents sentiment proportions in a dataset.
   - Uses Matplotlib to plot the sentiment categories as a pie chart.

Deployment

- Version control using **GitHub**.
- Streamlit Cloud is used for deployment.
- Dependencies managed through `requirements.txt`.