Team Members

1. Vhuvhwawo Mawela Masalesa (Group Leader)
2. Ondela Citywayo
3. Segai Bryton Mmpshika
4. Mthunzi Malebadi
5. Sinesipho Sibulo

Sentiment Analysis Dashboard Documentation

# Overview

The Sentiment Analysis Dashboard is a Streamlit-based web application that allows users to analyze the sentiment of textual data. It provides both single and multiple text input support, sentiment distribution visualizations, keyword extraction, polarity trends, and export options (CSV, JSON, PDF).

This dashboard is powered by the TextBlob library for Natural Language Processing (NLP), alongside matplotlib, pandas, and WordCloud for visualizations and data manipulation.

# Features

• Direct text input & file upload (.txt)

• Multi-class sentiment classification (positive, negative, neutral)

• Confidence score display for each sentiment prediction

• Keyword extraction to identify sentiment drivers

• Batch processing for multiple text entries

• Visualizations of sentiment distribution (Sentiment distribution pie chart, Polarity histogram, Line graph for polarity trends across sentences, Word cloud)

• Comparative analysis between different text sets

• Explanations of classification rationale

• Export results to CSV, JSON, and PDF formats

• Proper error handling for API failures and invalid inputs

# Architecture

The system architecture follows a linear flow:  
User Input → Preprocessing → API Call → Postprocessing → Dashboard UI  
  
The application uses Streamlit for UI, Python for logic and processing.

# Implementation Challenges

* **Sentence splitting:**

Simple period-based splitting can cause issues with abbreviations or ellipses. A more robust NLP parser (e.g., Panda) may improve accuracy.

* **Keyword extraction limitations:**

The extraction approach is basic and rule-based; performance may vary based on text complexity.

* **Streamlit PDF rendering:**

FPDF does not support dynamic layout well; adjustments had to be made for varying sentence lengths.

* **Polarity interpretation:**

TextBlob’s sentiment is limited by the quality of the rule-based lexicon. Some complex emotions may be misclassified.

# Installed Packages

Ensure the following Python packages are installed:

pip install streamlit textblob pandas matplotlib wordcloud fpdf

# How to Run the Dashboard

* Save your code into a file named sentiment\_dashboard.py
* Install the required packages using the command above
* In your terminal, navigate to the project folder and run:
* streamlit run sentiment\_dashboard.py
* The app will launch in your browser at **http://localhost:8501**

# Core Functions Explained

**analyze\_sentiment(text)**

* Uses TextBlob to analyze polarity and subjectivity
* Labels sentiment as Positive, Negative, or Neutral based on polarity

**generate\_wordcloud(text)**

* Creates a visual word cloud of keywords in the text

**get\_sentiment\_keywords(text)**

* Extracts key sentiment-bearing words (adjectives, nouns, verbs)

**create\_pdf\_report(df, overall\_summary, text\_content)**

* Dynamically generates a downloadable PDF report containing summary statistics and sentence-level analysis

# User Guide

**Example 1: Direct Text Input**

* Launch the app.
* In the sidebar, select “Direct Text Entry”.
* Paste a paragraph of text.
* Click outside the text area or press Enter.
* View the sentiment breakdown, graphs, and export options.

**Example 2: Uploading Multiple Files**

* Launch the app.
* In the sidebar, select “Upload Text File(s)”.
* Upload .txt files containing text data.
* Analyze each file’s sentiment, compare results, and download reports.
* Example input: “The product is amazing. However, the delivery was very slow.”

**Output:**

* Positive sentence: “The product is amazing”
* Negative sentence: “However, the delivery was very slow”
* Neutral summary with overall average polarity and subjectivity

# Limitations

- Potential misclassification of ambiguous or sarcastic texts  
- Limited support for non-English languages  
- Confidence scores may not always reflect model certainty

# Technology Stack

* Frontend: Streamlit
* NLP Engine: TextBlob
* Data Handling: pandas
* Visualization: matplotlib, WordCloud
* PDF Reporting: FPDF