Multilingual News Sentiment & Emotion Analyzer
My Learnings-Report

DAMARUK ABHISHEK VEDAGIRI
INSTRUCTOR-Naga Sai Sir
Date: 17-11-2025

INDEX

1. Introduction
2. What I Learned
3. Key Topics Covered
4. Project Summary
5. Challenges I Faced
6. Code/Output
7. Final Reflection

1. Introduction

This project focuses on analyzing bias in global news articles written in different languages. The system translates articles into English, identifies their sentiment and emotional tone, and compares the results. The goal was to understand how AI tools like Gemini and IBM NLU can be combined to study multilingual content more effectively.

2. What I Learned

- How multilingual text can be processed and analyzed using AI.
- Using Gemini for translation and IBM NLU for sentiment & emotion analysis.
- Building a complete workflow from backend processing to a user-facing UI.
- Visualizing results clearly through charts and dashboards.
- Debugging and structuring a real-world AI application.

3. Key Topics Covered

- Natural Language Processing (NLP)
- Translation (Gemini API)
- Sentiment and emotion analysis (IBM NLU)
- Data processing and visualization
- Streamlit UI development
- API integration and environment variables

4. Project Summary

In this project, I built a tool that analyzes sentiment and emotional bias in multilingual news articles. Users can upload text files, which are translated into English using Gemini. IBM NLU then extracts sentiment and emotions from each article. The Streamlit dashboard displays the results through graphs like sentiment bars, heatmaps, and radar charts. The system helps compare emotional tone across different regions and languages.

5. Challenges I Faced

- Handling API setup and authentication for both Gemini and IBM NLU.
- Fixing path issues, file loading errors, and import conflicts.
- Ensuring the UI could handle multiple file uploads without crashing.
- Maintaining clean integration between backend logic and the Streamlit interface.
- Managing environment variables safely while testing.

How I solved them:

By testing each component individually, adding debug logs, separating backend and UI code, and refining the file handling logic.

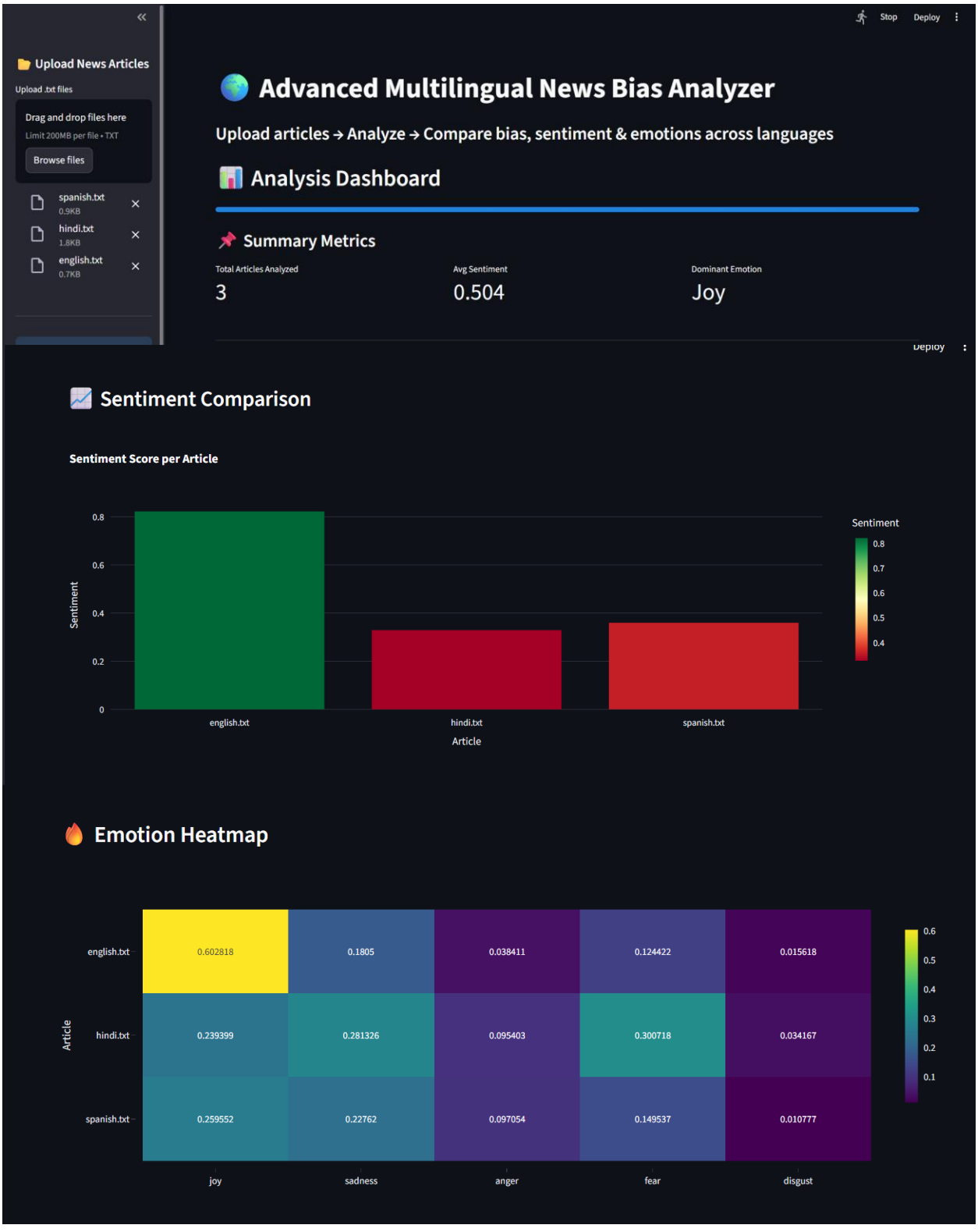
Code / Output

The project produces:

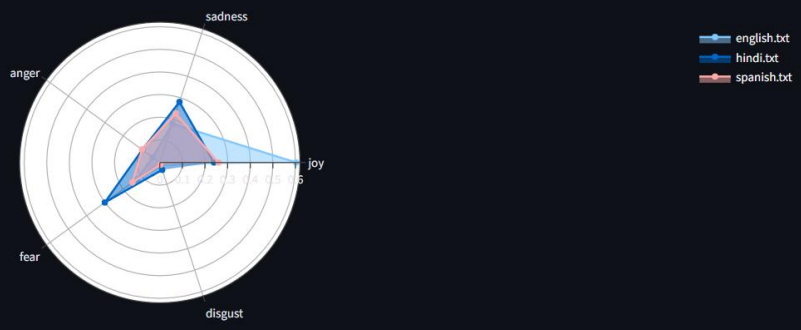
- Sentiment scores for each article
- Emotion breakdown (joy, sadness, anger, fear, disgust)
- Visual charts including sentiment bar charts, emotion heatmaps, and radar plots
- A downloadable CSV of all processed results

The code includes:

- Backend processing (code_files.py)
- Streamlit dashboard (app.py)
- Real-time translation and NLU analysis pipeline

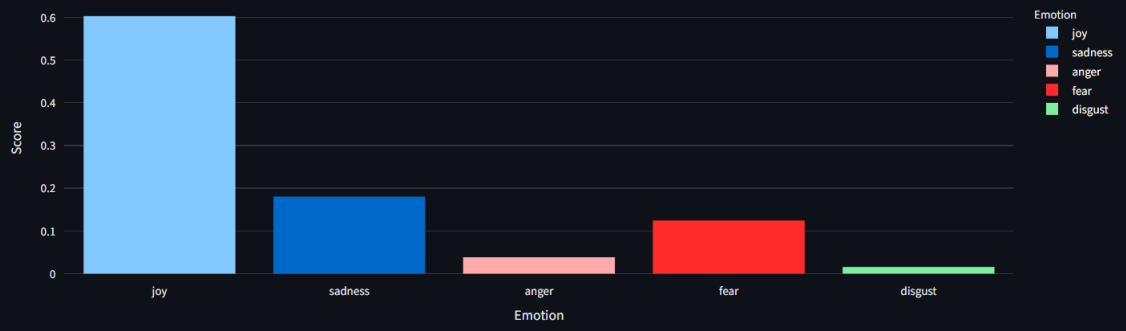

Upload News Articles
Upload .txt files
Drag and drop files here
Limit 200MB per file • TXT
Browse files
spanish.txt
0.9KB
hindi.txt
1.8KB
english.txt
0.7KB
Stop
Deploy
Advanced Multilingual News Bias Analyzer
Upload articles → Analyze → Compare bias, sentiment & emotions across languages
Analysis Dashboard
Summary Metrics
Total Articles Analyzed
3
Avg Sentiment
0.504
Dominant Emotion
Joy
Sentiment Comparison
Sentiment Score per Article
Sentiment
0.8
0.6
0.4
0.2
0
english.txt
hindi.txt
spanish.txt
Article
Emotion Heatmap
english.txt
hindi.txt
spanish.txt
Article
0.602818
0.1805
0.038411
0.124422
0.015618
0.239399
0.281326
0.095403
0.300718
0.034167
0.259552
0.22762
0.097054
0.149537
0.010777
joy
sadness
anger
fear
disgust
0.6
0.5
0.4
0.3
0.2
0.1

Emotion Radar Chart



Emotion Breakdown

Emotion Intensity



7. Final Reflection

This project gave me hands-on experience in building a real AI application. I learned how to combine multiple APIs, process multilingual text, and present insights visually. The dashboard helped me understand the importance of making data easy to interpret. Moving forward, I want to explore more advanced NLP techniques like topic modeling and deploy apps like this on cloud platforms.