Decoding emotions through sentiment analysis of social media conversations

Student name:Dharunkumar.S

Register Number:732323106009

Institution:SSM COLLEGE OF ENGINEERING

Department: Electronic communication engineering

Date of submission:23.04.2025

Problem Statement

Social media users express a wide range of emotions in their posts, but due to the informal and massive nature of the data, it's difficult to analyze these emotions manually. This project aims to build a system that uses sentiment analysis to automatically detect and classify emotions in social media conversations, helping to better understand public mood and opinions in real-time.

Objectives of the project

1.To collect social media data related to specific topics or hashtags.

2. To preprocess and clean the collected text data.

3 .To perform sentiment analysis and classify posts as positive, negative, or neutral.

4.To detect deeper emotions such as joy, anger, sadness, and surpprise using NLP models.

Scope of the Project

This project focuses on analyzing emotions in social media conversations using sentiment analysis and Natural Language Processing (NLP). It is limited to textual data collected from platforms like Twitter. The analysis will cover basic sentiment classification (positive, negative, neutral) as well as deeper emotion detection (joy,

anger, sadness, etc.). The results will be visualized to understand public sentiment trends over time. This project can be extended in the future to include multilingual support, image/video sentiment, and real-time monitoring.

Data Source

The primary data source for this project is Twitter, accessed using the Twitter API via the Tweepy library in Python. Tweets are collected based on specific keywords or hashtags relevant to trending topics or events. Each tweet includes text content, timestamp, user information (anonymized), and location data (if available). The data is used solely for academic and research purposes.

High -level Methodology

1. Data Collection

Tweets are collected using the Twitter API with the help of the Tweepy library.

Keywords and hashtags related to the topic of interest are used (e.g., #MentalHealth, #Elections).

Metadata such as tweet text, date, time, and user location are stored for analysis.

2. Data Cleaning

Remove unnecessary elements such as:

URLs, hashtags, mentions, emojis, special characters.

Convert text to lowercase for uniformity.

Remove stop words and perform tokenization and lemmatization using NLTK or spaCy.

3. Exploratory Data Analysis (EDA)

Analyze word frequencies and common hashtags.

Check the distribution of tweet lengths, sentiment scores, and time-based trends.

Use word clouds and bar charts to understand dominant themes and words.

4. Feature Engineering

Convert text data into numerical format using:

TF-IDF (Term Frequency-Inverse Document Frequency)

Word embeddings (e.g., BERT embeddings)

Add derived features such as sentiment score or tweet length.

Prepare data for input into machine learning models.

Model Building

1. Data Preparation

Cleaned and preprocessed tweet text is converted into numerical format using:

TF-IDF Vectorization (for traditional models).

Word Embeddings from pre-trained models like BERT (for deep learning models).

2. Model Selection

A. For Sentiment Analysis (Positive, Negative, Neutral)

Traditional ML Models:

Logistic Regression

Support Vector Machine (SVM)

Random Forest

These models are effective with TF-IDF features and work well for binary or multi-class classification.

Deep Learning / NLP Models:

BERT (Bidirectional Encoder Representations from Transformers)

RoBERTa (Robustly optimized BERT)

These models are fine-tuned for text classification and achieve higher accuracy on complex emotional language.

B. For Emotion Classification

Used pre-trained models from HuggingFace, such as:

bhadresh-savani/bert-base-go-emotion

Fine-tuned on datasets like GoEmotions (27 emotion categories)

3. Model Training

Training-Validation Split: 80/20 ratio

Loss Function: Cross-Entropy Loss (for multi-class classification)

Optimizer: AdamW

Metrics Tracked: Accuracy, F1-score, Precision, Recall

4. Hyperparameter Tuning

Used GridSearchCV (for ML models) or adjusted learning rate, batch size, and epochs (for transformer models).

5. Model Evaluation

Use classification models like Logistic Regression, SVM, or fine-tuned BERT for sentiment and emotion classification.

Split the dataset into training and test sets.

Evaluate models using metrics such as:

Accuracy

Precision

Recall

F1-Score

Confusion Matrix

6. Visualization & Interpretation

Use matplotlib, seaborn, and Plotly to visualize:

Sentiment distribution

Emotion frequencies

Trends over time (e.g., how emotions change across days or events)

Word clouds for each sentiment or emotion

7. Deployment

Create a simple web app or dashboard using Streamlit or Flask to:

Input a keyword or hashtag

Show live sentiment and emotion analysis results

Display interactive graphs and word clouds

Tools and Technologies

1. Programming Language

Python

Python is widely used in data science and NLP due to its simplicity and rich ecosystem.

2. Notebook / IDE

Jupyter Notebook or Google Colab

Used for interactive coding, data visualization, and documentation in one place.

Colab supports cloud-based execution (no setup needed).

Jupyter works locally and integrates with many Python environments.

3. Libraries

a. Data Collection

Tweepy – To connect with the Twitter API and stream tweets.

b. Data Processing

Pandas – For data manipulation and handling CSV/JSON formats.

NumPy – For numerical operations.

c. NLP and Sentiment Analysis

NLTK / spaCy – For text cleaning, tokenization, lemmatization.

TextBlob / VADER – For basic sentiment classification.

HuggingFace Transformers (e.g., BERT, RoBERTa) – For deep emotion classification.

d. Feature Engineering

Scikit-learn – For TF-IDF vectorization and model evaluation metrics.

e. Visualization

Matplotlib, Seaborn – For static plots and graphs.

Plotly, WordCloud – For interactive and thematic visualizations.

4. Optional Tools for Deployment

Streamlit – Quickly build a web app to showcase the sentiment analysis in real-time.

Flask – A lightweight web framework to create APIs and integrate with frontend dashboards.

Team Members and Roles

1.Dinesh.S

Responsibilities

Oversee the entire project lifecycle,manages time line and coordinates tasks

2.Dharunkumar.S

Responsibilities

Processing and analysing text data,handle language normalisation and sentiment-specific preprocessing.

3.Deepika.P

Responsibilities

Perform Exploratory Data Analysis (EDA) to understand data trends,

Visualize emotion distribution using charts and graphs

4.Dhinakaran.S

Responsibilities

Design and build a user-friendly interface (using Streamlit, Flask, or similar),

Integrate the trained emotion detection model into the application