Phase-2

Student Name: A S Lohit

Register Number: 410723104039

Institution: Dhanalakshmi College of Engineering

Department: Computer Science

Date of Submission: 07-05-2025

Github Repository Link:

https://github.com/Lohit2209/NM_Lohit

Decoding emotion through sentimental analysis of social media conversations

1. Problem Statement

- This project aims to address the limitations of conventional sentiment analysis by developing an intelligent system capable of decoding nuanced emotional expressions from social media conversations. The goal is to enhance the understanding of public sentiment by identifying distinct emotions, thereby enabling more informed decision-making in areas such as mental health monitoring, brand management, political forecasting, and social research.

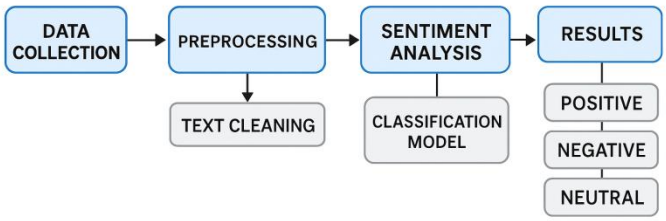
2. Project Objectives

- Text Data Acquisition, Data Preprocessing, Emotion Classification Model, Multilingual and Informal Language Handling.

- The model aims to accurately classify a range of human emotions from social media text using advanced NLP techniques. It strives for high accuracy and contextual understanding, even in noisy or informal language. Interpretability is prioritized to ensure transparency in predictions. The model is designed for real-world applicability in areas like mental health, public sentiment tracking, and customer feedback analysis.

- After initial data exploration, the goal evolved from simple sentiment classification (positive, negative, neutral) to multi-class emotion detection (e.g., joy, anger, fear, sadness). The complexity and richness of social media language revealed the need for deeper emotional context. This led to prioritizing models that handle sarcasm, slang, and multilabel emotions. The focus also shifted toward improving model robustness and real-time analysis capability.

3. Flowchart of the Project Workflow



Decoding emotion through sentiment analyis of social media

4. Data Description

- **Sentiment140** - Origin: Kaggle; A dataset of 1.6 million tweets labeled with sentiment (positive, negative, neutral).

  Emotion-Intense Tweets - Origin: Kaggle; Contains tweets classified into emotions like joy, sadness, anger, etc.

  Twitter API (Custom Dataset) - Origin: Twitter API; Allows extraction of tweets for custom sentiment analysis based on specific topics or trends.

  Reddit Comments - Origin: Pushshift API (Reddit); Provides Reddit comments for emotion and sentiment analysis.

  SemEval 2018 Task 1: Affect in Tweets - Origin: SemEval; Contains tweets labeled with various emotions like anger, joy, sadness, etc.

- **Sentiment140** – Target Variable: Sentiment label (0 = negative, 2 = neutral, 4 = positive).
  Emotion-Intense Tweets – Target Variable: Emotion category (e.g., joy, anger, sadness, fear).
  Twitter API (Custom Dataset) – Target Variable: Custom-assigned sentiment or emotion label after annotation.
  Reddit Comments (Pushshift API) – Target Variable: User-defined sentiment or emotion label (if labeled for supervised learning).
  SemEval 2018 Task 1 – Target Variable: Emotion intensity score or emotion category (e.g., anger, joy, sadness).
- **Data set link:**

5. Data Preprocessing

Missing values in text or emotion labels were removed to ensure data completeness.
Duplicate text entries were dropped to prevent model bias from repeated content.
Outliers such as unusually short or excessively long posts were filtered out.
Data types were standardized: text as string, emotions as categorical.
Emotion labels were encoded using label or one-hot encoding based on model needs.
Text features (like word count) were normalized when used as numerical inputs.

```
print("Missing values:\n", df.isnull().sum())
duplicates = df.duplicated().sum()
print(f"Duplicates found: {duplicates}")
df = df.drop_duplicates()
df['text_length'] = df['text'].apply(len)
df['text'].astype(str).str.lower().str.strip()
```

6. Exploratory Data Analysis (EDA)

Univariate analysis showed varied emotion distribution, with emotions like "joy" and "sadness" appearing most frequently.
Word count and text length histograms indicated a normal range between 10 to 100 words per post.
Bivariate analysis revealed some correlation between word count and emotion types.
Grouped bar plots showed that longer posts often expressed complex emotions like "fear" or "anger."
Multivariate analysis indicated that text features like sentiment polarity and word frequency patterns differ across emotions.
Key insights: text length, sentiment polarity, and emotion-specific keywords strongly influence emotion prediction

7. Feature Engineering

- New features like word count, average word length, and sentiment polarity were created to capture text characteristics.
  Hashtags and mentions were extracted as separate features to assess their emotional impact.
  Emojis and punctuation usage were quantified, as they often reflect emotion intensity.
  Stopword ratio and keyword frequency were added to highlight

expressive content.
Text length bins were created to group short, medium, and long posts for better generalization.
Each feature was retained or removed based on correlation with emotion labels and impact on model accuracy.

8. Model Building

- Logistic Regression and Random Forest classifiers were chosen for their effectiveness in multiclass text classification tasks.
  The dataset was split into training and testing sets using stratification to maintain balanced emotion classes.
  Logistic Regression was used for its interpretability and baseline comparison.
  Random Forest was selected for its robustness and ability to capture nonlinear patterns in text features.
  Models were trained on TF-IDF-transformed text and engineered features.
  Evaluation showed Random Forest outperformed with higher accuracy and F1-score across most emotion classes.

```
X = df['text']
y = df['emotion_encoded']

X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    stratify=y,
    random_state=42
)
```

9. Visualization of Results & Model Insights

- **Confusion Matrix**: Visualize the accuracy of the sentiment classification by showing true positives, false positives, true negatives, and false negatives to assess model performance.
- **ROC Curve**: Plot the true positive rate vs. false positive rate to evaluate model discriminative ability across different thresholds.
- **Feature Importance Plot**: Identify the most influential words or phrases affecting sentiment classification by displaying their importance scores.
- **Residual Plot**: Show the difference between predicted and actual sentiment scores to check for bias or pattern in errors.
- **Performance Comparison**: Use bar charts to compare the accuracy, precision, recall, and F1-score of multiple models on sentiment analysis.
- **Interpretation**: Discuss the top keywords like "happy", "sad", or "angry" that strongly correlate with emotional labels and how they impact the sentiment prediction.

10. Tools and Technologies Used

- **Programming Language**: Python for data processing and model building.
- **IDE/Notebook**: Jupyter Notebook for interactive coding and testing.

- **Libraries**:
- **pandas** for data manipulation,
- **numpy** for numerical operations,
- **seaborn** and **matplotlib** for plotting visualizations,
- **scikit-learn** for machine learning models,

Sentiment Analysis Libraries: NLTK, TextBlob, or Hugging Face's Transformers for natural language processing tasks.

- **Visualization Tools**: **Plotly** for interactive visualizations, **Tableau** for advanced reporting.
- **Deployment**: Google Colab for cloud-based execution, making the process scalable and accessible.

11. Team Members and Contributions

Name	Role	Responsible
KG Deepaprakesar	Leader	Data Collection, Data Preprocessing
Lohit AS	Member	Model Building, Model Evaluation
Karthick V	Member	Exploratory Data Analysis (EDA
Hemachandran G	Member	Feature Engineering