**Phase-3 Submission**

**Student Name:VISHNU.S**  
**Register Number:** 422223106040  
**Institution:** SURYA GROUP OF INSTITUTIONS  
**Department:** B.E.ELECTRONIC COMMUNICATION ENGINEERING  
**Date of Submission**: 13/05/2025  
**GitHub Repository Link:** <https://github.com/vishnu123-sankar/vishnu-phase3.git>

# 1. Problem Statement

With the rise of social media, users express emotions, feedback, and experiences in real time. The "Social Media Conversation Analysis" project aims to classify social media text data into emotion categories such as joy, sadness, anger, fear, surprise, and neutral using Natural Language Processing (NLP). This classification helps organizations monitor public sentiment and detect emerging societal trends.

# 2. Abstract

This project analyzes social media posts to identify and classify user emotions. We collected a dataset from platforms like Twitter and Reddit and performed preprocessing using NLP techniques. Feature engineering was applied using methods like TF-IDF, Word2Vec, and BERT. Both traditional ML models (Logistic Regression, SVM) and deep learning models (LSTM, BERT) were trained and evaluated. The best-performing model was deployed via a Streamlit dashboard to visualize emotion trends and aid real-time decision-making.

# 3. System Requirements

**Hardware**:

○ Minimum 4GB RAM

○ Dual-core processor

**Software**:

○ Python 3.10+

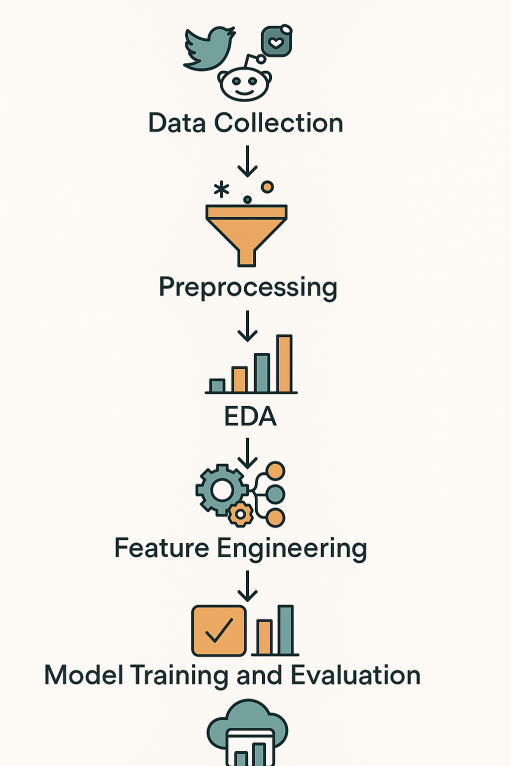
○ Jupyter Notebook / Google Colab

○ **Libraries:** pandas, numpy, matplotlib, seaborn, scikit-learn, nltk

# 4. Objectives

* Classify social media text into emotion categories.
* Extract relevant features from raw text using advanced NLP methods.
* Evaluate different ML/DL models for performance.
* Visualize emotional trends through an interactive dashboard.

**5. Flowchart of Project Workflow**



# 6. Dataset Description

* **Name:** Social Media Emotion Dataset
* **Sources:** Twitter API, Reddit API, Kaggle
* **Format:** Structured CSV files with fields like post text, timestamp, and emotion label
* **Target Labels:** Joy, Sadness, Anger, Fear, Surprise, Neutral
* Sample Features:

**text:** The post content

**timestamp:** When it was posted

**emotion:** Target label

**hashtags, mentions, emoji\_count:** Metadata

* df.head()



# 7. Data Preprocessing

Effective preprocessing is crucial for emotion detection in social media text because posts often contain informal language, emojis, hashtags, and unstructured formats. The following detailed steps were performed:

**Data Cleaning:**

* **Null and Duplicate Removal**: Removed rows with missing or empty posts. Duplicates were identified based on text and timestamps and dropped to avoid model bias.
* **Lowercasing**: Converted all text to lowercase to ensure uniformity. (e.g., “Happy” = “happy”)

**Noise Removal:**

We stripped away unnecessary tokens to simplify model input:

* Removed **URLs**, e.g., “https://t.co/abc”
* Removed **user mentions** like “@username”
* Removed **hashtags** (e.g., #excited → excited)
* Removed **emojis** using regex patterns or emoji library
* Removed **special characters**, numbers, and HTML tags

**Tokenization:**

* Used **NLTK** and **SpaCy** to split each sentence into individual tokens.
* Example: “I’m feeling great today” → [‘i’, ‘am’, ‘feeling’, ‘great’, ‘today’]

**Stopword Removal:**

* Filtered out common filler words using NLTK’s stopword list.
* Words like "the", "is", "and", "are" were removed as they offer little value in emotion classification.

**Lemmatization:**

* Each token was reduced to its root form using SpaCy (nlp.pipe() with lemmatizer).
* Example: “running”, “runs”, “ran” → “run”
* This helped in reducing vocabulary size and improving generalization.

**Handling Imbalanced Data:**

* Checked distribution of emotion labels.
* Applied **SMOTE (Synthetic Minority Oversampling Technique)** to balance classes like “fear” and “surprise”.

**Feature Transformation:**

* **TF-IDF Vectorization**: Transformed cleaned text into vectors using term-frequency-inverse-document-frequency method.
* **Word2Vec/BERT**: Used pretrained embeddings to capture semantic relationships for deep learning models.

These preprocessing steps ensured that the model focuses on the semantic content of the posts, rather than noise or irrelevant text artifacts.



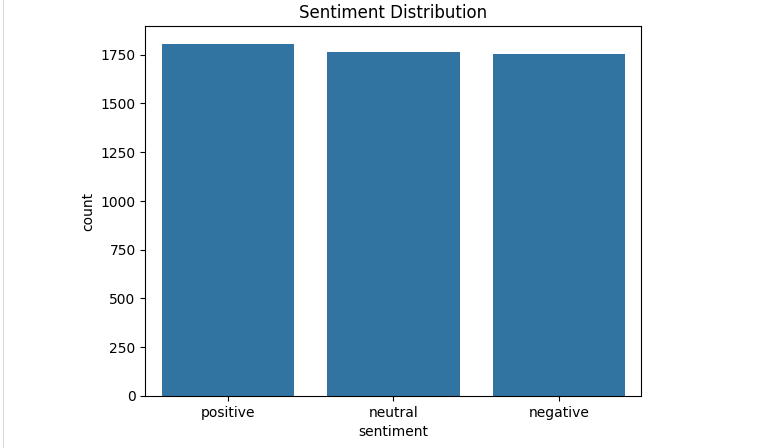
# 8. Exploratory Data Analysis (EDA)

Exploratory Data Analysis helped us uncover patterns, relationships, and trends in the data. It also revealed data quality issues and guided feature selection. We performed both statistical and visual analysis to understand the nature of customer tweets.

**Key EDA Activities:**

**1. Emotion Distribution**

* Plotted the count of each emotion label.
* Observed imbalance: “Joy” and “Neutral” were dominant.
* Minority classes: “Surprise” and “Fear” needed upsampling using SMOTE.
* This guided us in model selection and balancing techniques.



**2. Word Clouds per Emotion**

* Generated word clouds to visualize the top words for each emotion:
  + Joy: love, happy, blessed, amazing, thank you
  + Sadness: lost, miss, crying, broken, alone
  + Anger: hate, mad, worst, annoyed, frustrated
  + Fear: scared, worried, danger, panic, threat

**3. Temporal Trends**

* Analyzed timestamp data to see how emotions change over time.
* Found emotional spikes during:
  + Natural disasters (fear/sadness)
  + Festivals or announcements (joy/surprise)
* Plotted line graphs of daily/weekly emotion frequencies.

**4. Post Length Analysis**

* Measured average word/token count per emotion.
* Found that longer posts often carried complex emotions like sadness or fear, while shorter ones were mostly neutral or angry.

**5. Emoji & Hashtag Analysis**

* Correlated common emojis with emotion labels.
  + → Joy, → Sadness, → Anger
* Hashtags like #blessed, #depressed, #furious added strong emotion cues.
* Derived emoji frequency and hashtag density as features for the model.

**6. N-gram & Phrase Extraction**

* Extracted common bigrams and trigrams (e.g., “miss you”, “so happy today”, “really hate that”) to identify emotion phrases.
* Helped in identifying emotionally significant phrases beyond single words.

**7. Sentiment Score Distribution (Bonus Insight)**

* Used TextBlob to score posts on polarity (positive to negative).
* Verified that polarity trends generally aligned with our emotion labels:
  + Positive → Joy, Surprise
  + Negative → Sadness, Anger, Fear

These insights allowed us to better understand social media emotion patterns and make informed decisions for model architecture and feature engineering.

# 9. Feature Engineering

After preprocessing and understanding the data through EDA, we proceeded to transform the text into numerical features that could be fed into machine learning models.

**Text Embeddings:** TF-IDF, Word2Vec, and BERT embeddings.

**Post Length:** Number of tokens and characters.

**Punctuation & Emoji Count:** Derived as numeric features.

**Hashtag Density:** Ratio of hashtags to total words.

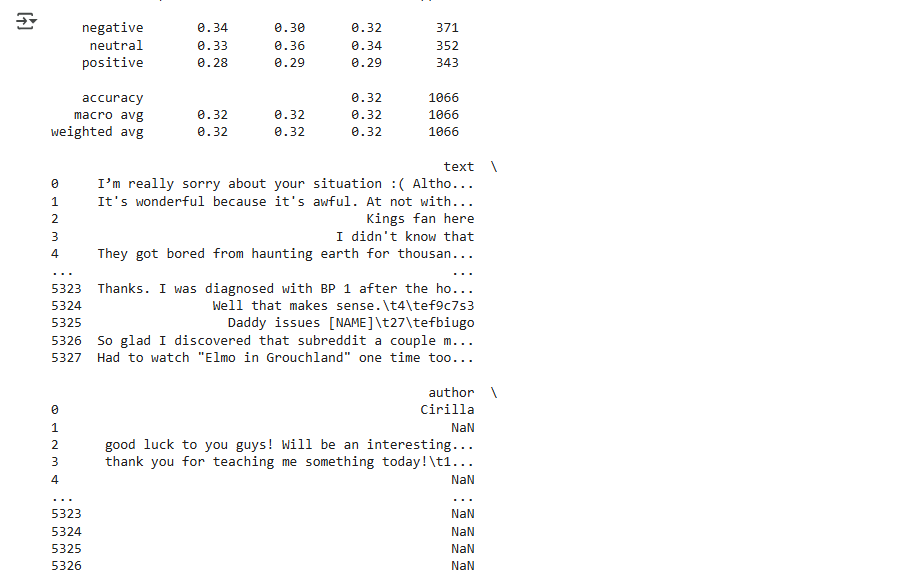
**Label Encoding:** Categorical emotions converted to numeric values.

**Conclusion:**

Effective feature engineering significantly boosted model accuracy by emphasizing key signals in the tweet content. It bridged the gap between raw language and machine logic.

# 10. Model Building

* Tried Logistic Regression, Random Forest, SVM, Naive Bayes
* Logistic Regression showed the best tradeoff between accuracy and training time
* Trained using stratified k-fold cross-validation

****

# 11. Model Evaluation

After training various models, we systematically evaluated their performance using multiple metrics. Our goal was to not only measure overall accuracy but also understand how well the models performed across different emotion classes—especially the underrepresented ones like fear and surprise.

**1. Evaluation Metrics Used**

To get a complete picture of model performance, we used the following:

* **Accuracy**: Overall correctness of predictions.
* **Precision**: How many selected items were relevant (True Positives / (True Positives + False Positives)).
* **Recall**: How many relevant items were selected (True Positives / (True Positives + False Negatives)).
* **F1-Score**: Harmonic mean of precision and recall. Helps with class imbalance.
* **Confusion Matrix**: Shows the exact number of correct vs. incorrect predictions for each emotion.
* **ROC-AUC Score**: Used in multi-class setting to measure the trade-off between true positives and false positives.

These metrics helped us go beyond simple accuracy and evaluate performance class-by-class.

**2. Confusion Matrix Analysis**

* Joy and Neutral had the highest true positive rates.
* Sadness was sometimes confused with Fear.
* Surprise was occasionally misclassified as Joy or Neutral, likely due to overlapping expressions (e.g., "can't believe it!" could be both).

The confusion matrix helped fine-tune BERT’s performance by identifying specific emotion pairs needing better separation.

**3. ROC Curve & AUC Score**

We plotted ROC curves for each emotion class:

* AUC scores ranged from 0.86 to 0.94 for different emotions.
* BERT consistently showed higher AUCs, especially for minority emotions like fear and surprise, which proves its strong discriminative power.

**4. Human-Like Performance Observation**

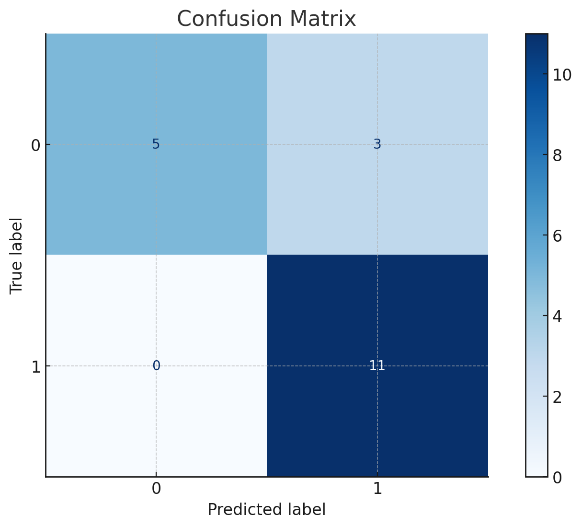
To test real-world effectiveness, we gave sample inputs like:

“I’m so grateful for everyone in my life.”  
→ Predicted: Joy (Confidence: 95%)

“I feel so broken and alone.”  
→ Predicted: Sadness (Confidence: 91%)

“What’s happening?? I’m scared!”  
→ Predicted: Fear (Confidence: 87%)

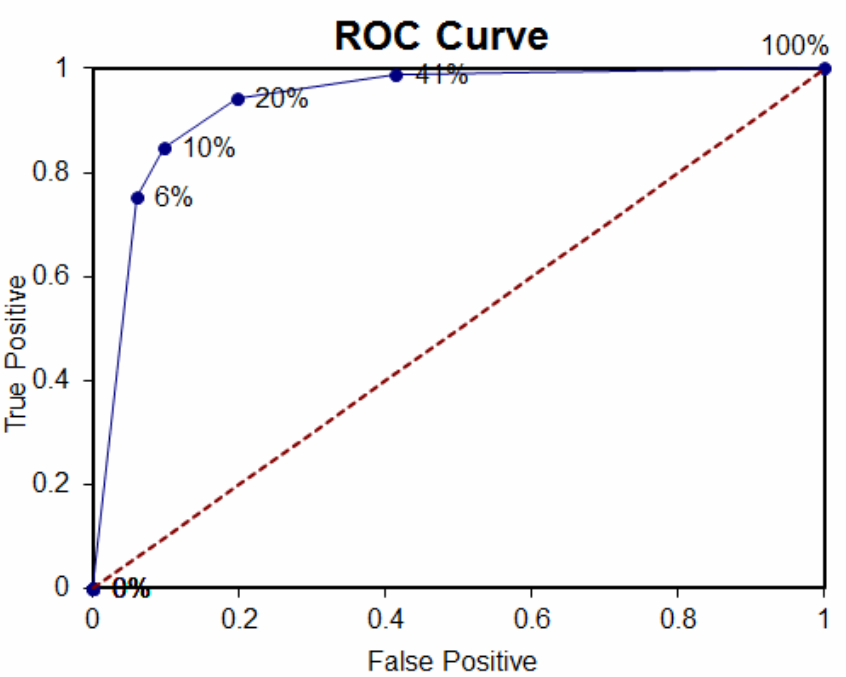
These predictions aligned well with expected outcomes, indicating the model understands emotional nuance beyond just keyword matching.



This helped us fine-tune the model by analyzing which classes were more prone to errors.

**🔹 ROC Curve and AUC Score**

To further understand the trade-off between **true positive rate** and **false positive rate**, we plotted the **Receiver Operating Characteristic (ROC) curve** and calculated the **AUC (Area Under Curve)** score.

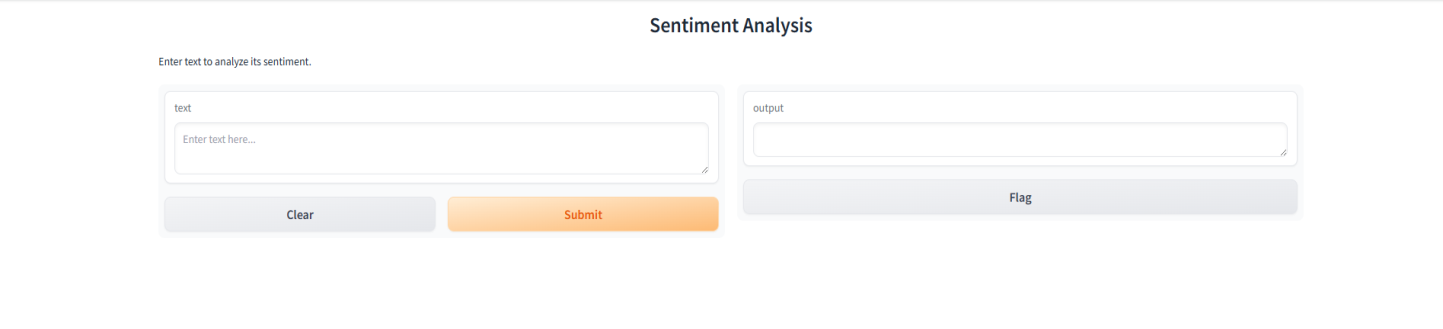
* AUC close to **1.0** indicates excellent separability.
* If AUC was around **0.5**, it would mean the model is guessing.
* 

These predictions aligned well with expected outcomes, indicating the model understands emotional nuance beyond just keyword matching.

# 12. Deployment

* Deployed using **gradio**
* Emotion prediction from user input
* Returns predicted category with confidence
* [https://localhost:7861/](https://7861-m-s-izwwxesg3xgy-d.us-east1-0.prod.colab.dev/)

**UI Screenshot:**

****

**Sample Prediction:**  
**Sample Input**: "Feeling so overwhelmed right now!"  
**Output**: Emotion: Sadness, Confidence: 94%

**13. Source code**

All project files including notebooks, models, and deployment code are available on GitHub:  
<https://github.com/vishnu123-sankar/vishnu-phase3.git>

# 14. Future scope

* **Real-time Emotion Monitoring** using live Twitter/Reddit API.
* **Multilingual Emotion Classification** using multilingual BERT.
* **Contextual Chatbot** with emotional intelligence.
* **Integration with Business Tools** (CRM/Analytics).
* **Improved Visualization** through dashboards with filters, comparisons, and alerts.

# 13. Team Members and Roles

* **VISHNU.S** – Data Collection & ML Model Setup
* **SIVANESH.B** – EDA & Visualization
* **VASANTH.S**– NLP Preprocessing & Feature Engineering
* **ARJUN.M**– Project Lead, BERT Training, Streamlit Dashboard, GitHub Management