TEXT ANALYTICS PROJECT

Project Objective

- Build a **machine learning model** to predict emotions from textual data.
- Analyze text data using **exploratory data analysis (EDA)**.
- Apply **TF-IDF feature extraction** and **logistic regression**
- models.
- Address **class imbalance** and **overfitting** using regularization and class weighting.

Dataset Overview

- Rows: 21,459, Columns: 2 (Text, Emotion)
- **No missing values** in the dataset.
- **Target Classes**: anger, fear, happy, love, sadness, surprise.

```
(21459, 2)
                                                Text  Emotion
0                            i didnt feel humiliated  sadness
1  i can go from feeling so hopeless to so damned...  sadness
2   im grabbing a minute to post i feel greedy wrong    anger
3  i am ever feeling nostalgic about the fireplac...     love
4                                i am feeling grouchy    anger
```

```
print(df.isnull().sum())
```

```
Text       0
Emotion    0
dtype: int64
```

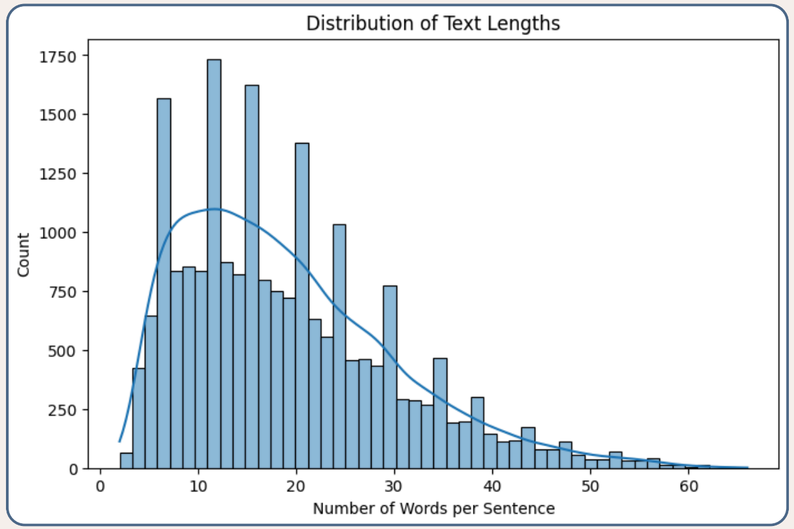
Exploratory Data Analysis

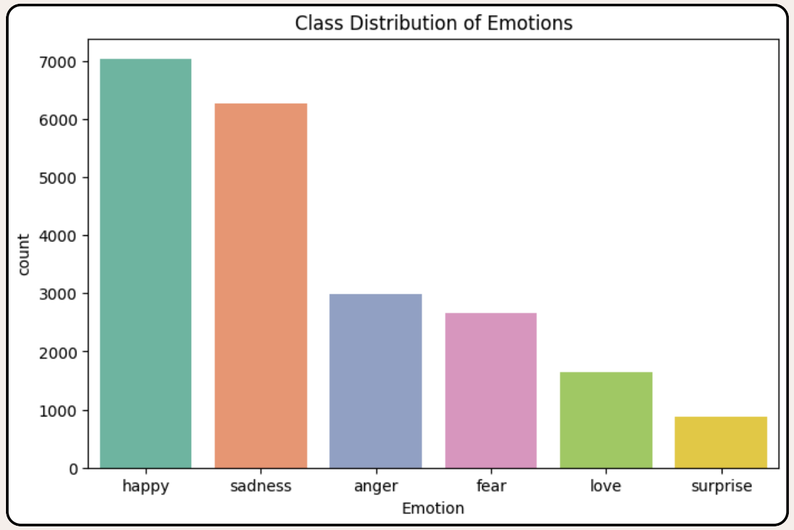
Class Distribution:

- Most frequent: happy, sadness
- Least frequent: surprise

Text Length Analysis:

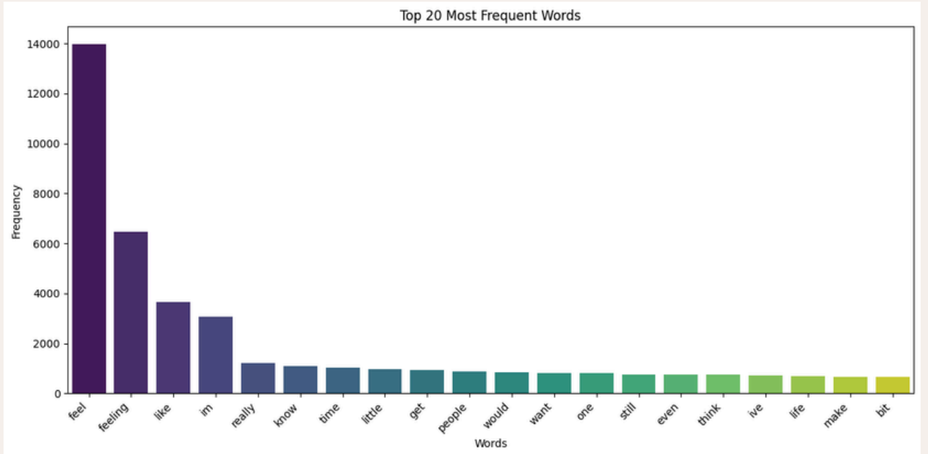
- Average sentence length varies.
- Helps understand text complexity for model training.

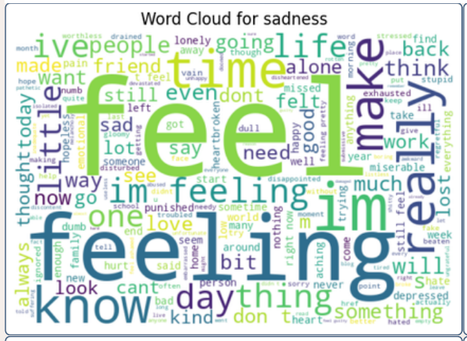


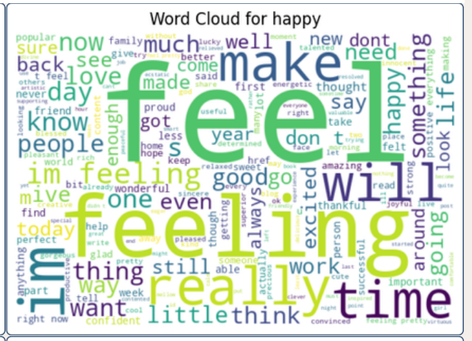


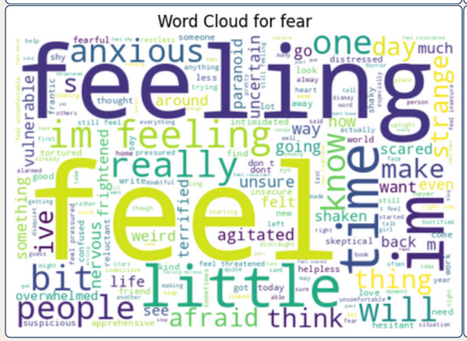
Word Frequency & Word Clouds

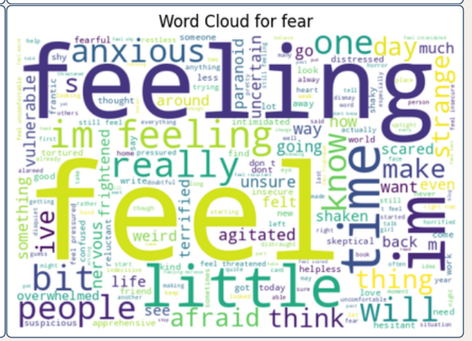
- Extracted **Top 20 Most Frequent Words** using Counter.
- Generated **word clouds per emotion** to visualize commonly used words.











Data Preprocessing

Text cleaning steps:

- Convert to lowercase
- Remove URLs and punctuation
- Remove stopwords (nltk)
- Resulting column: Cleaned_Text
- Prepare text for *TF-IDF vectorization.

```
0                                [didnt, feel, humiliated]
1    [go, feeling, hopeless, damned, hopeful, aroun...
2    [im, grabbing, minute, post, feel, greedy, wrong]
3    [ever, feeling, nostalgic, fireplace, know, st...
4                                      [feeling, grouchy]
Name: Cleaned_Text, dtype: object
```

Feature Extraction

- Used **TF-IDF Vectorizer** to convert text into numeric features.
- **Shape of TF-IDF matrix:** (21459, 19092)
- Captures **word importance** across the dataset.

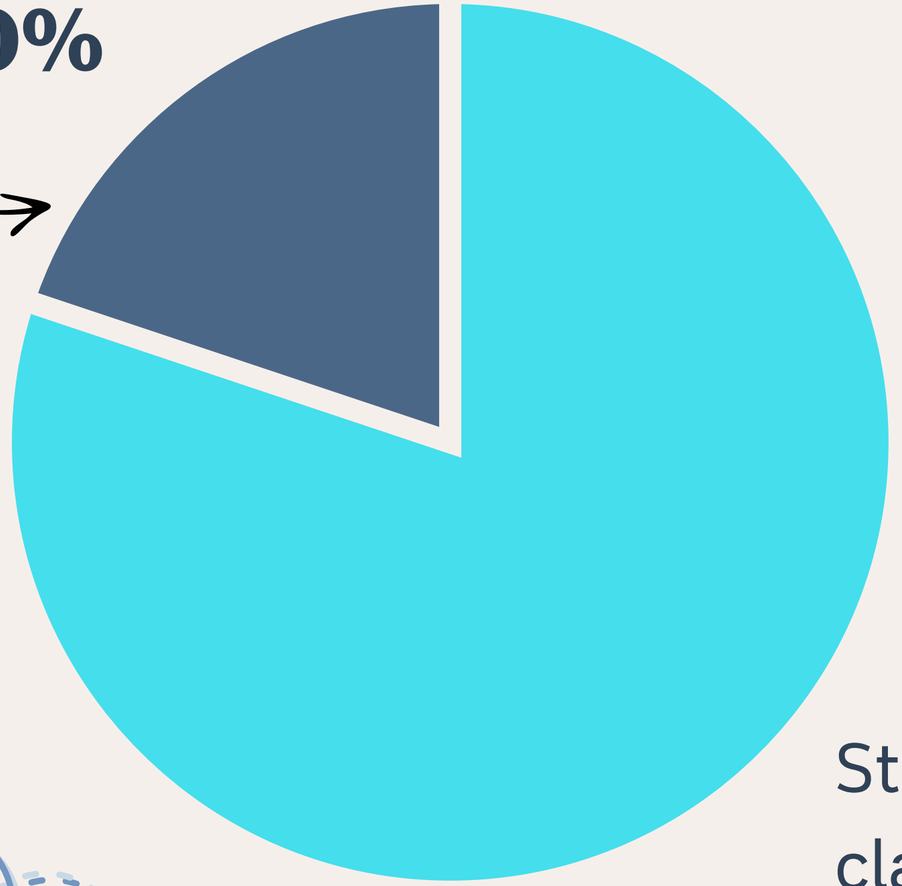
```python
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf_vectorizer = TfidfVectorizer()
X_tfidf = tfidf_vectorizer.fit_transform(df["Cleaned_Text"])

print("Shape of the TF-IDF matrix:", X_tfidf.shape)
```

```
Shape of the TF-IDF matrix: (21459, 19092)
```

Train Test Split

Split dataset into:

Testing: 20%



Training: 80%

Stratified split to maintain class distribution.

Model Training - Logistic Regression

Models Built:

1. Without Regularization
2. With L2 Regularization
3. With Class Weights
4. L2 Regularization + Class Weights (Best Model)

Goal: Compare performance & handle overfitting and class imbalance

Model Evaluation

Metrics:

- Precision, Recall, F1-Score, Accuracy
- Evaluated on Training and Testing datasets

Observation:

- Training accuracy ~95%
- Testing accuracy ~86–88%
- Weighted + L2 Regularized model improves minority class predictions (love, surprise).

ssification Report for Training Data (L2 Regularization and Weighte

	precision	recall	f1-score	support
anger	0.93	0.97	0.95	2394
fear	0.94	0.96	0.95	2122
happy	0.99	0.91	0.95	5623
love	0.79	0.99	0.88	1313
sadness	0.98	0.94	0.96	5012
surprise	0.82	1.00	0.90	703
accuracy			0.94	17167
macro avg	0.91	0.96	0.93	17167
ighted avg	0.95	0.94	0.95	17167

ssification Report for Testing Data (L2 Regularization and Weighted

	precision	recall	f1-score	support
anger	0.85	0.88	0.87	599
fear	0.84	0.83	0.84	530
happy	0.94	0.89	0.91	1406
love	0.74	0.91	0.82	328
sadness	0.94	0.89	0.91	1253
surprise	0.66	0.86	0.75	176
accuracy			0.88	4292
macro avg	0.83	0.88	0.85	4292
ighted avg	0.89	0.88	0.88	4292

Classification Report for Training Data (L2 Regularization):

	precision	recall	f1-score	support
anger	0.97	0.94	0.95	2394
fear	0.96	0.92	0.94	2122
happy	0.93	0.98	0.95	5623
love	0.95	0.83	0.89	1313
sadness	0.95	0.98	0.97	5012
surprise	0.97	0.78	0.86	703
accuracy			0.95	17167
macro avg	0.95	0.90	0.93	17167
weighted avg	0.95	0.95	0.95	17167

Classification Report for Testing Data (L2 Regularization):

	precision	recall	f1-score	support
anger	0.88	0.78	0.83	599
fear	0.86	0.76	0.81	530
happy	0.83	0.96	0.89	1406
love	0.90	0.62	0.74	328
sadness	0.87	0.94	0.91	1253
surprise	0.82	0.47	0.59	176
accuracy			0.86	4292
macro avg	0.86	0.75	0.79	4292
weighted avg	0.86	0.86	0.85	4292

Classification Report for Training Data (No Regularization):

	precision	recall	f1-score	support
anger	0.97	0.94	0.95	2394
fear	0.96	0.92	0.94	2122
happy	0.93	0.98	0.95	5623
love	0.95	0.83	0.89	1313
sadness	0.95	0.98	0.97	5012
surprise	0.97	0.78	0.86	703
accuracy			0.95	17167
macro avg	0.95	0.90	0.93	17167
weighted avg	0.95	0.95	0.95	17167

Classification Report for Testing Data (No Regularization):

	precision	recall	f1-score	support
anger	0.88	0.78	0.83	599
fear	0.86	0.76	0.81	530
happy	0.83	0.96	0.89	1406
love	0.90	0.62	0.74	328
sadness	0.87	0.94	0.91	1253
surprise	0.82	0.47	0.59	176
accuracy			0.86	4292
macro avg	0.86	0.75	0.79	4292
weighted avg	0.86	0.86	0.85	4292

Model Insights

- **L2 Regularization**: reduces overfitting
- **Class Weights**: improves predictions for minority classes
- **Most accurately predicted emotions**: happy, sadness
- **Less accurately predicted**: surprise, love

Model Deployment

- Model saved using joblib: **emotion_model.pkl**
- Can predict emotions on new text samples:

Examples:

```
I am so excited about my new job! → Predicted Emotion: happy
This is making me very sad today → Predicted Emotion: sadness
I feel nervous before the exam → Predicted Emotion: fear
```

Key Takeaways & Future works

- Text preprocessing and feature extraction are critical for NLP tasks.
- TF-IDF with logistic regression is effective for emotion detection.
- Using regularization and class weighting improves model robustness.
- Potential for real-time deployment to analyze user sentiments.

- Explore deep learning models like LSTM, BERT for improved accuracy.
- Incorporate emoji and slang handling.
- Build a web app for real-time emotion detection.

THANK YOU!