📊 Twitter Sentiment Analysis – Logistic Regression Model

✅ Project Summary Report

🔍 Objective

This project aims to build a **Sentiment Analysis model** to classify tweets as either **Positive** or **Negative** based on the textual content. It helps businesses understand customer feedback, social trends, or product opinions in real time using machine learning.

📁 Dataset

The dataset includes the following columns:

- ID – Unique identifier for each tweet
- Topic – Subject/topic related to the tweet (optional in modeling)
- Sentiment – The labeled sentiment (Positive/Negative)
- Review – Text content of the tweet or customer review

🛠️ Tools & Libraries Used

- **Python**
- **Scikit-learn**
- **Pandas, Numpy**
- **Matplotlib, Seaborn**
- **TfidfVectorizer**

🧹 Data Preprocessing

The following steps were applied to clean and prepare the text data:

- Converted all reviews to **lowercase**
- **Removed punctuation marks** and special characters

- **Removed stopwords** (like "the", "is", "and", etc.)
- Tokenized the text and normalized spacing
- Used **TF-IDF Vectorization** (ngram_range=(1,2), stop words removed)

🧠 Model Training

- Algorithm: **Logistic Regression**
- Vectorization: **TF-IDF**
- Train-Test Split: 80/20
- Training set transformed using TfidfVectorizer, and the same was applied to test and new data

✅ Evaluation Results

Metric	Value
Accuracy	88.5%
Precision (Positive)	90%
Precision (Negative)	87%
F1-Score (Overall)	~88%

📌 Confusion Matrix:

```
[[4208  369]
 [ 621 3477]]
```

🔎 Model Testing Examples

Input Text	Predicted Sentiment
"you are bad boy"	❌ *Positive* (Incorrect)

Input Text	Predicted Sentiment
"I really love this phone!"	✅ Positive
"Worst experience ever!"	✅ Negative

Note: Early testing revealed bias in prediction due to TF-IDF mismatch during vectorization. Issue was resolved by using the same fitted vectorizer during both training and prediction.

📁 Deliverables

- Clean and modular Python code (.py or Jupyter notebook)
- PDF Report with evaluation results
- GitHub repository containing:
  - Code
  - Preprocessing functions
  - Vectorizer and model training steps
  - Test examples and predictions

💡 Key Learnings

- Importance of consistent TF-IDF vectorization across training and inference
- Handling text preprocessing for noisy social media content
- Logistic Regression as an effective baseline for sentiment classification