Approach Used

The sentiment analysis model is based on **Logistic Regression** with **TF-IDF vectorization**. The approach involves the following steps:

1. **Text Preprocessing:**
    - Converted text to **lowercase**.
    - Removed **digits, special characters, and punctuation**.
    - Eliminated **stopwords** using NLTK's stopwords list.
2. **Feature Engineering:**
    - Used **TF-IDF (Term Frequency-Inverse Document Frequency)** to convert text into numerical feature vectors.
    - Limited the vocabulary to **10,000 features** to improve efficiency.
3. **Model Training:**
    - Split the dataset into **80% training and 20% testing**.
    - Trained a **Logistic Regression** model with `max_iter=1000` to ensure convergence.

Challenges Faced

- **Text Noise:** Removing irrelevant words without losing contextual meaning was crucial.
- **Feature Selection:** The choice of `max_features=10000` in TF-IDF needed tuning to balance performance and computational cost.
- **Imbalanced Data (If Any):** If sentiment classes were imbalanced, it could affect model generalization.

Model Performance & Improvements

1. **Performance Metrics:**
    - **Accuracy: 89.16%**
    - **F1-Score: 89.40%**
2. **Observations:**
    - The model achieves **high accuracy** in sentiment classification.
    - Logistic Regression is an effective baseline but may struggle with **complex linguistic patterns**.
3. **Possible Improvements:**
    - **Use Word Embeddings:** Implement **Word2Vec** or **pretrained embeddings (e.g., GloVe, FastText)** for better semantic understanding.
    - **Try Deep Learning Models:** LSTMs or **transformer-based models like BERT** could improve performance.
    - **Hyperparameter Tuning:** Adjust parameters like `C` (regularization) and `max_features` in TF-IDF for optimization.