# **Cloud-Based NLP with Transformers: Sentiment Analysis API**

## **Objective**

The goal of this project was to design and implement a cloud-based NLP solution using transformer models to analyze sentiment in text. The system was built using the Hugging Face Transformers library and deployed as a REST API using Flask.

## **Project Implementation**

### **1. Dataset Preparation**

* The dataset consisted of customer reviews with sentiment scores.
* Data was cleaned by removing duplicates and missing entries.
* Sentiments were categorized into three classes:
  + **Negative** (scores 1-2)
  + **Neutral** (score 3)
  + **Positive** (scores 4-5)
* The dataset was split into **training (80%)** and **testing (20%)** sets.

### **2. Model Selection and Training**

* We used **DistilBERT**, a lightweight transformer model, for sentiment classification.
* The text was tokenized using the **AutoTokenizer** from Hugging Face.
* A **custom PyTorch dataset** was created to handle tokenized text and labels.
* The model was fine-tuned on the dataset for **3 epochs** with batch sizes optimized for memory efficiency.
* Training was done using the **Trainer API**, with performance metrics like accuracy, precision, recall, and F1-score.

### **3. Model Evaluation**

* The model achieved a good balance of accuracy and precision across all sentiment categories.
* The best-performing model was saved for deployment.

### **4. Deployment as a Cloud API**

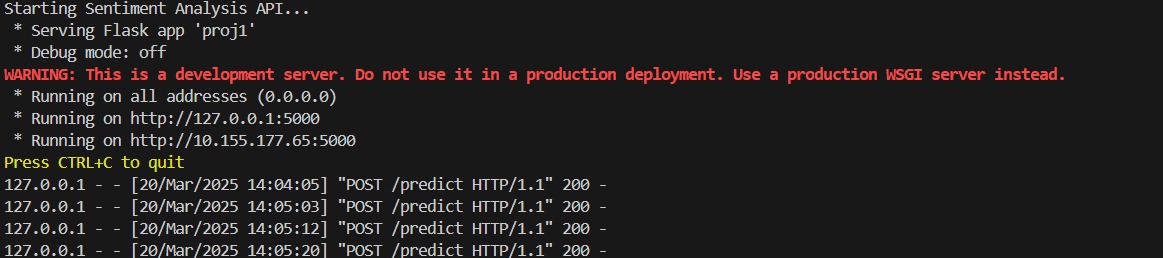
* The trained model was deployed using **Flask**, allowing users to send text data via an API endpoint (/predict).
* The API takes a JSON input and returns the sentiment classification.
* It was hosted on **localhost** for testing.

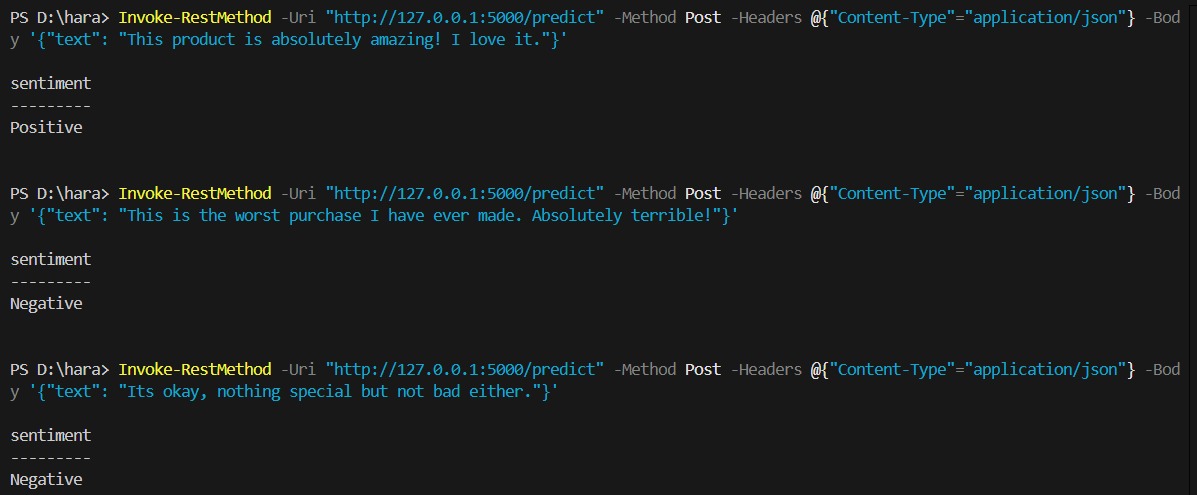
## **Results**

The API successfully classified sentiment in real-time:

|  |  |
| --- | --- |
| **Input Text** | **Predicted Sentiment** |
| "This product is absolutely amazing! I love it." | **Positive** |
| "This is the worst purchase I have ever made." | **Negative** |
| "It's okay, nothing special but not bad either." | **Neutral** |

### **Figure 1: Flask API Running and Processing Requests**





**Figure 2: Sentiment Analysis API Predictions**

These results confirm that the model correctly identifies different sentiments, making it a useful tool for analyzing customer feedback.

## **Future Improvements**

* **Fine-tuning on larger datasets** to improve accuracy.
* **Enhancing model interpretability** to provide reasons for sentiment classification.
* **Optimizing inference speed** for real-time applications.

This project successfully demonstrates the power of transformer-based NLP models in a cloud-based setting, making sentiment analysis efficient and scalable.