Sentiment Analysis Model Documentation

# 1. Project Overview

This project performs sentiment classification using a transformer-based model. It categorizes text into one of four sentiment classes using the Hugging Face Transformers library. The training dataset includes 964 examples, tokenized and fed into a fine-tuned DistilBERT model.

# 2. Model Used

Model: distilbert-base-uncased  
DistilBERT is a lightweight transformer model derived from BERT that provides strong performance with fewer resources. It is used here for sequence classification with a classification head on top.

# 3. Tokenization

Text samples were tokenized using the model's tokenizer. Padding and truncation ensured consistent input shape. The tokenized dataset was converted to PyTorch tensors using `tokenized\_dataset.set\_format('torch')`.

# 4. Training Pipeline

Training was done using Hugging Face's `Trainer` API with the following configuration:  
- Learning Rate: 2e-5  
- Batch Size: 8  
- Epochs: 3  
- Weight Decay: 0.01  
- Logging Steps: 10  
- Evaluation Strategy: 'epoch'  
- Save Strategy: 'epoch'  
- EarlyStoppingCallback with patience=2  
- Best model selection based on eval\_loss

# 5. Accuracy Improvement Steps

After training, we observed class imbalance affecting performance, especially for class 1.  
To improve model accuracy, the following strategies were applied or recommended:  
- Apply class weights in loss function  
- Augment training data (back translation, synonym replacement)  
- Switch to stronger base models like `bert-base`, `roberta-base`, or `deberta-v3-small`  
- Hyperparameter tuning (batch size, learning rate, epochs)  
- Use confusion matrix for misclassification analysis

# 6. Evaluation

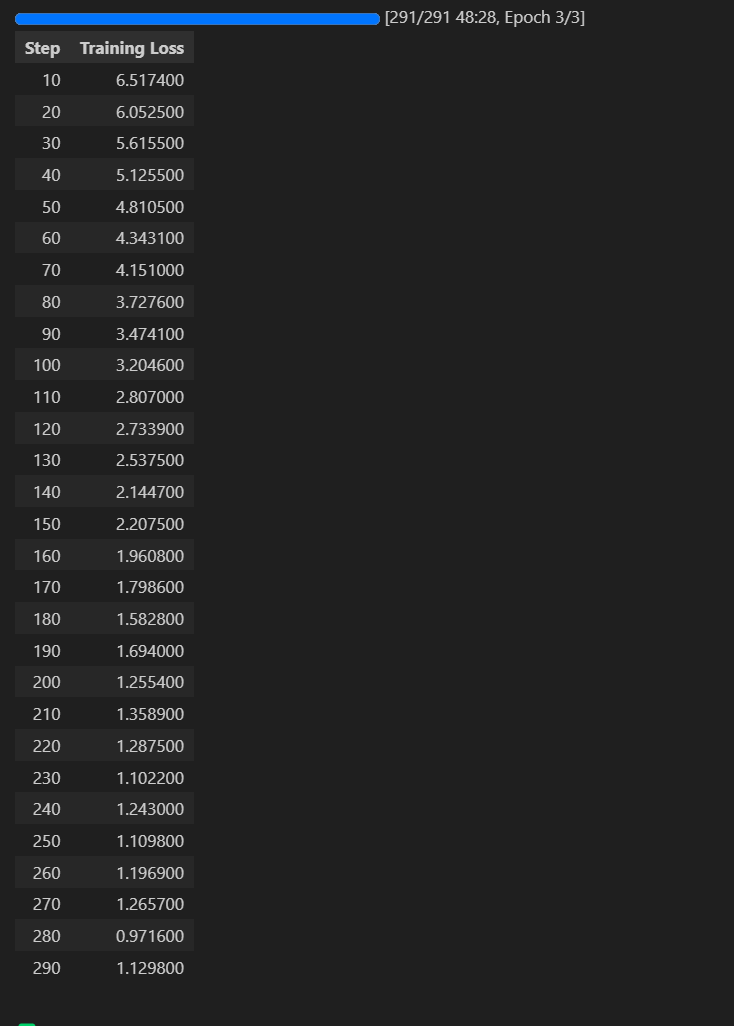
The model achieved an accuracy of 81%. Class-wise performance showed high precision for class 1, but poor recall. The classification report and training loss indicated room for optimization.

# 7. Prediction Function

The trained model is used to predict sentiment on new input text. The function uses softmax on model logits and returns the label mapped using a `label\_mapping.pkl` file.

# 8. Training Loss Curve

Below is the training loss observed over 290 steps:

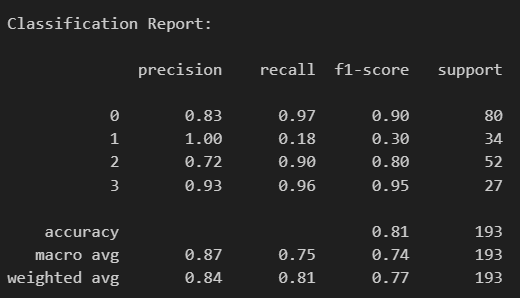




## 4.1. How This Project Works

This project classifies user-provided text into predefined sentiment categories using a fine-tuned DistilBERT model. It includes the following steps:  
1. Input text is tokenized using a pretrained tokenizer.  
2. Tokenized input is passed to a transformer model (DistilBERT).  
3. The model predicts logits, which are converted into probabilities using softmax.  
4. The class with the highest probability is chosen as the final prediction.  
5. The numerical label is mapped to a human-readable class using a label mapping file.

## 4.2. Model Accuracy Summary



After training, the model achieved an overall accuracy of 81% on the validation set. Class-wise performance was as follows:  
- Class 0: Precision 0.83, Recall 0.97  
- Class 1: Precision 1.00, Recall 0.18 (Low recall due to class imbalance)  
- Class 2: F1-score of 0.80  
- Class 3: F1-score of 0.95  
Macro F1-score: 0.74  
Weighted F1-score: 0.77

## 4.3. How to Further Improve Accuracy

To enhance model performance:  
- Use a more powerful model like `bert-base-uncased`, `roberta-base`, or `deberta-v3-small`  
- Add more labeled training data  
- Balance the dataset or use class weights in the loss function