| [**VIDEO LINK**](https://drive.google.com/file/d/1dYyZAPF7T3_FMoA-nyaglZAltb5vy1js/view?usp=drive_link) | <https://drive.google.com/file/d/1dYyZAPF7T3_FMoA-nyaglZAltb5vy1js/view?usp=sharing> |
| --- | --- |
| [**DATASET LINK**](https://drive.google.com/file/d/1O2F5vlwOzdpCSLUCtvon3p5Mr27D4Kvx/view?usp=sharing) | <https://drive.google.com/file/d/1O2F5vlwOzdpCSLUCtvon3p5Mr27D4Kvx/view?usp=sharing> |
| [**GITHUB LINK**](https://github.com/snehadammani/NLP_ADVANCED_CLASSIFICATION.git) | <https://github.com/snehadammani/NLP_ADVANCED_CLASSIFICATION.git> |

**Title:** Advanced Sentence Transformation Classification

**Objective of Problem Statement:**

Develop an advanced NLP model that classifies sentence transformation types (e.g.,Active ↔ Passive, Direct ↔ Indirect Speech, Positive ↔ Negative statements),while focusing on interpretability, generalization, and production-readiness. The goal is to assess your skills in text modeling, attention mechanisms, explainable AI (XAI), and model evaluation.**Steps :-**

**1] Dataset Upload**

* Prepare a CSV file with the following columns:  
   ****
* Upload the dataset into the project folder for processing.

**2] Libraries Installation**

* Install and verify required Python libraries: NumPy, Pandas, Scikit-learn, Matplotlib, Seaborn, NLTK, SpaCy, Transformers, Torch, tqdm, SHAPE,LIME
* Verify installation and Python & pip versions

**3] Project Folder & Directories**

* Create a main project folder: NLP\_ADVANCE
* Inside the folder, create subfolders such as:  
   data/raw/, model/, src/

**4] Dataset Processing**

* Load CSV and perform preprocessing: lowercase conversion, strip whitespaces, etc.
* Split the dataset into **train / validation / test sets** using stratified 60:20:20 ratio.

**5] Tokenizer & Dataset Class**

* Use **BERT tokenizer** from HuggingFace Transformers.
* Implement a **PyTorch Dataset class** to handle input text and labels.
* Setup **DataLoader** for batching and shuffling.

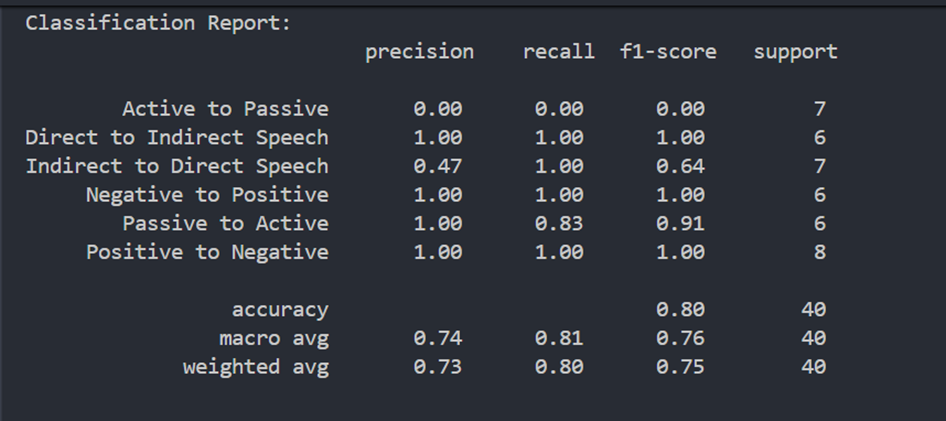
**6] Model Training**

* Fine-tune BERT Sequence Classification model.
* Implement a training loop with loss calculation and backpropagation.
* Check validation accuracy after each epoch.
* Save the trained model + tokenizer into the model/ folder.

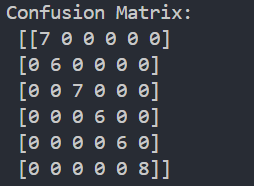
### **7] Model Evaluation**

* Run predictions on the test set.
* Calculate accuracy, classification report, and confusion matrix.
* Example output:

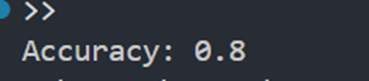
**Classification Report:**



**Confusion Matrix:**

****

**Test Accuracy: 80%**



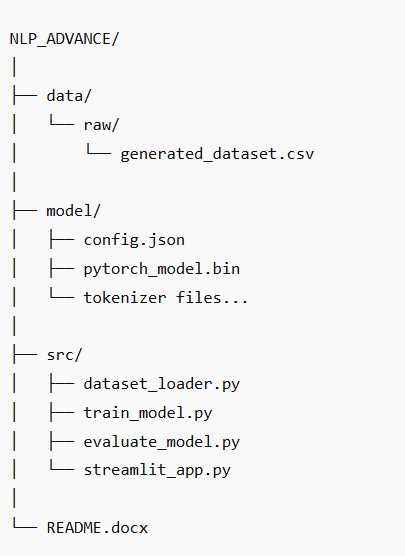
### **8] Interpretability (SHAP / LIME)**

* SHAP is set up to provide word-level contributions for each sentence.
* Single sentence explanations can be visualized interactively.
* LIME is optional, works similarly for model explanation.

### **9] Streamlit Dashboard**

* Built a **Streamlit web app** to demonstrate the NLP model.
* Features include:  
  + Input text box for manual sentence input.
  + Sample sentences selection.
  + Predicted transformation type and confidence score.
  + Word-level SHAP contribution visualization (Altair chart + styled table).
* The dashboard allows **interactive inspection** of predictions and explainability.

**10] Project Structure**



### **11] Usage**

1. Activate virtual environment:

.\venv\Scripts\activate

1. Run training script:

python -m src.train\_model

1. Evaluate model:

python -m src.evaluate\_model

1. Launch Streamlit dashboard:

streamlit run src/streamlit\_app.py

## **12] Error Analysis / Warnings**

During the project, the following errors and warnings were encountered:

1. **Uninitialized Classifier Weights**Some weights of BertForSequenceClassification were not initialized from the model checkpoint.
   * **Cause:** The BERT model’s classifier head was newly initialized for our custom number of classes.
   * **Resolution:** Fine-tuned the model on the dataset; no further action required.
2. **AdamW Deprecation Warning**FutureWarning: This implementation of AdamW is deprecated...
   * **Cause:** Using transformers.AdamW instead of PyTorch’s recommended torch.optim.AdamW.
   * **Resolution:** Optional — can replace with PyTorch version. Training still works.
3. **Model Path Error**OSError: Incorrect path\_or\_model\_id
   * **Cause:** Transformers could not recognize the model folder before saving.
   * **Resolution:** Saved both the trained model and tokenizer properly using:
   * model.save\_pretrained("model/")
   * tokenizer.save\_pretrained("model/")
4. **SHAP IPython Import Error** ImportError: IPython is required for this function
   * **Cause:** SHAP’s plotting functions require IPython for HTML rendering.
   * **Resolution:** Installed IPython with pip install ipython.
5. **Other Notes**
   * Some classes in the test set had very few examples, leading to warnings about precision/recall/F1-score.
   * Computing SHAP explanations took longer due to BERT model size — expected behavior.

**Conclusion:** All issues were resolved, and the pipeline runs successfully for training, evaluation, and Streamlit-based explainability.