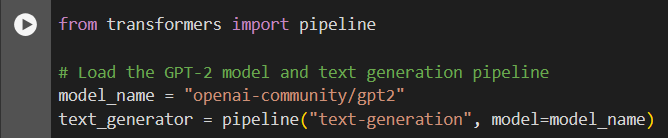
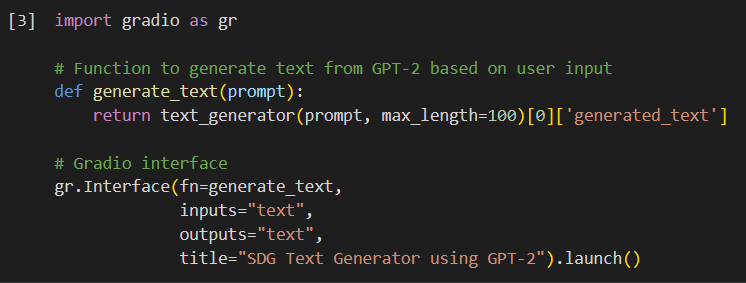
**Q1: Text Generation Application**

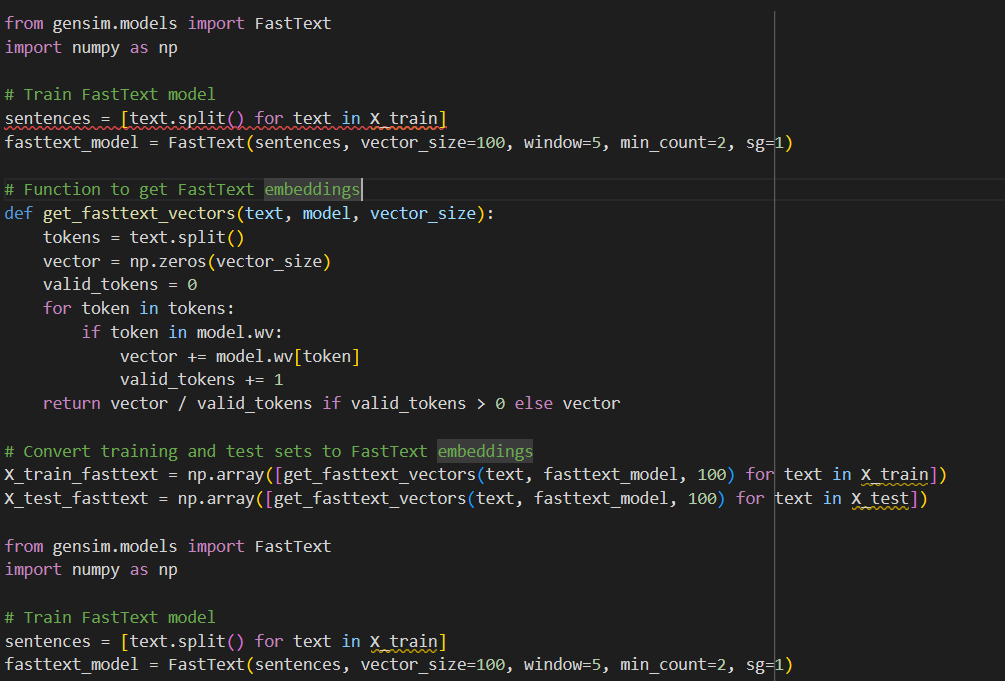
****

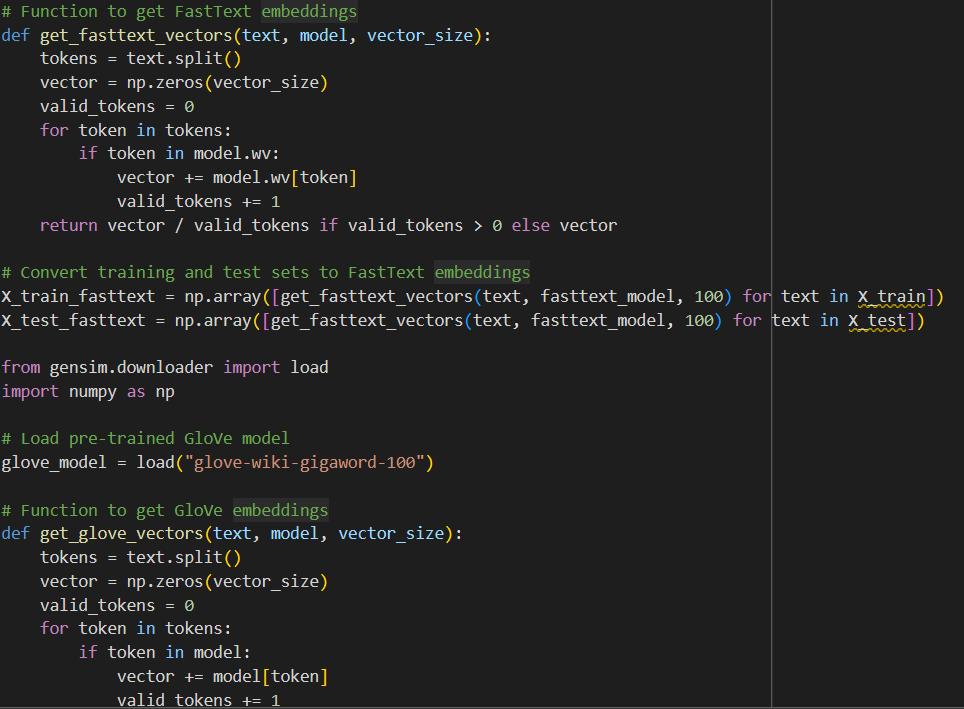
****

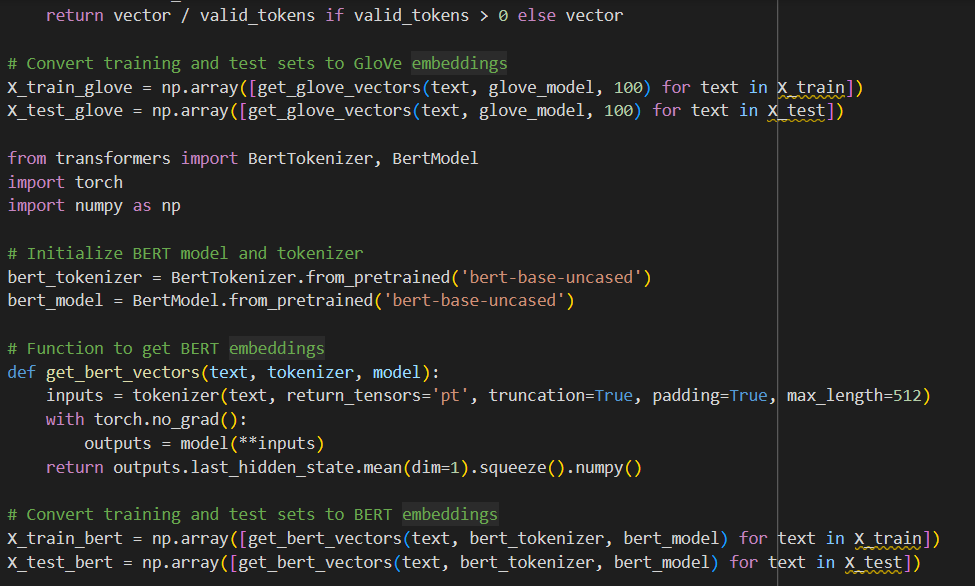
****

The openai-community/gpt2 model demonstrated limitations in generating accurate and meaningful content when prompted with SDG-related queries. The generated outputs often lacked the coherence and relevance needed for the task, indicating that the model may not be ideal for applications requiring precise and topic-specific text generation.

**Q2: Natural Language Processing Project**

****

****

****

This code demonstrates how different text embeddings are applied to a dataset, specifically for transforming text data into numerical vectors using \*\*FastText\*\*, \*\*GloVe\*\*, and \*\*BERT\*\* models.

1. FastText Embeddings:

- A FastText model is trained on the training dataset, where each sentence is split into tokens (words).

- A function is provided to calculate the FastText embeddings by averaging the vectors of tokens found in the model’s vocabulary. If a token is not found, it is ignored.

- The training and test sets are then converted into FastText embeddings using this function.

2. GloVe Embeddings:

- A pre-trained GloVe model is loaded using `gensim`'s downloader.

- A similar function is defined to obtain GloVe embeddings, which averages the vectors of valid tokens from the GloVe model.

- The training and test sets are converted into GloVe embeddings using this function.

3. BERT Embeddings:

- The BERT tokenizer and model are initialized using the `transformers` library.

- A function is created to obtain BERT embeddings by tokenizing the text and passing it through the BERT model. The output embeddings are averaged across all tokens in the sequence.

- The training and test sets are converted into BERT embeddings using this approach.

In each case, the text data from the training and test sets is transformed into a numerical representation that can be used in further machine learning tasks, such as classification or clustering.