Code Understanding Report

Generated: 2025-05-06 16:16:56

This report presents automated insights based on large language models and code analysis tools.

File: app.py

Summary

- This Python function is designed to accept a list of text strings as input and return them as a sparse tf-idf matrix, which allows to represent text data in a format that can be used for further analysis. Tfidf is short for term frequency-inverse document frequency, and is a statistical measure used to evaluate how important a word is to a document in a collection or corpus.
- This function calculates cosine similarity between two documents. Cosine similarity is a measure used to measure how similar the documents are by measuring the cosine of the angle between two vectors of an inner product space of the documents.

This similarity measure is often used when document similarity is required, or for natural language processing tasks, such as information retrieval. This function takes two documents as input and returns the cosine similarity between them.

Input: doc1, doc2 - are two documents. Output: cosine_similarity - value of cosine similarity between doc1 and doc2.

Assum - This code checks for plagiarism between students using the cosine similarity between their text vectors. It works by finding the similarity score between each pair of students' text vectors, and then storing the results in a set, which automatically removes duplicate entries.

The code is part of a larger system that checks for plagiarism between all pairs of students. The `s_vectors` is a list of tuples, where each tuple contains a student's ID and a text vector representation of their work.

The function `check_plagiarism` will run within the scope of the larger

Docstring

- ### Code: def vectorize(Text): return TfidfVectorizer().fit_transform(Text).toarray()

Docstring:

This function takes a list of text documents and returns a matrix of TF-IDF features.

Parameters: - Text: A list of strings, where each string is a document.

Returns: - A sparse - ### Code: def similarity(doc1, doc2): return cosine_similarity([doc1, doc2])

Docstring:

This function calculates the cosine similarity between two documents.

Parameters: - doc1: The first document. - doc2: The second document.

Returns: The cosine similarity between the two documents. - ### Code: def check*plagiarism(): global s*vectors for student*a, text*vector*a in s*vectors: new*vectors = s*vectors.copy() current*index = new*vectors.index((student*a, text*vector*a)) del *new*vectors[current*index] for student*b, text*vector*b in new*vectors: sim*score = similarity(text*vector*a, text*vector*b)[0][1] student*pair = sorted((student*a, student*b)) score = (student*pair[0], student*pair[1], sim*score) plagiarism*results.add(score) return plagiarism*results

Docstring:

This function checks for plagiarism in a list of text vectors. It does this by comparing each text vector with every other text vector in the list. The similarity between the two text vectors is calculated using the `similarity` function

Code Quality

Tool: `pylint`
Issues: 0`

```
text ************* Module tmpgg6_mivi C:
\Users\nmoha\AppData\Local\Temp\tmpgg6_mivi.py:21:8: W0602: Using
global for 's_vectors' but no assignment is done (global-
variable-not-assigned)
```

Conclusion

This codebase is designed to generate a sparse tf-idf matrix for text data and calculate cosine similarity between two documents. The tf-idf matrix is a valuable data structure for natural language processing tasks, while the cosine similarity metric provides a measure of how similar the documents are to one another. Together, these features allow for a range of tasks, such as document similarity and topic extraction, in text processing.