TP: Information Retrieval Models
(Term Document Matrix and Vector Space Model)

Problem Set:

Learn how to create and interpret a Term Document Matrix (TDM) for a set of documents. And apply the Vector Space Model to calculate document similarity using cosine similarity.

Problem 1: Create a Term Document Matrix

Task: Create a Term Document Matrix for a small set of documents.

Instructions:

1. Choose the following three sample documents:
    - Document 1: " Data science combines statistics, computer science, and domain knowledge."
    - Document 2: " Machine learning algorithms can analyze large datasets and make predictions."
    - Document 3: " Data visualization helps in interpreting complex data and communicating insights."
2. Write a function to tokenize each document (split into words) and count the frequency of each term.
3. Construct the Term Document Matrix (TDM) and print it.

Problem 2: Visualize the Term Document Matrix

Task: Display the Term Document Matrix in a readable format.

Instructions:

1. Using the TDM created in Problem 1, format the matrix into a table.
2. Ensure that the rows represent documents and the columns represent terms.
3. Print the TDM with appropriate labels for documents and terms.

Problem 3: Implement TF-IDF

Task: Calculate TF-IDF weights for the terms in the TDM.

Instructions:

1. Using the TDM from Problem 1, write a function to calculate the TF-IDF for each term in each document.

$$\text{TF-IDF}(t,d) = \text{TF}(t,d) \times \log\left(\frac{N}{\text{DF}(t)}\right)$$

2. Display the TF-IDF matrix.

Problem 4: Calculate Cosine Similarity

Task: Compute the cosine similarity between a query and the documents.

Instructions:

1. Define a query, for example: "data science algorithms".
2. Write a function to convert the query into a vector based on the terms in the TF-IDF matrix.
3. Implement the cosine similarity formula:

$$\text{Cosine Similarity} = \frac{A \cdot B}{\|A\| \|B\|}$$

   where A is the query vector and B is the document vector.

4. Rank the documents based on their cosine similarity to the query and print the results.

Problem 5: Advanced Query Processing and Cosine Similarity

Task: Preprocess documents and queries, then calculate cosine similarity with enhanced text normalization techniques.

Instructions:

1. Define multiple queries, for example:

- Query 1: "data scientist"
- Query 2: "machine learn"
- Query 3: "visualization of data"

2. Implement the following steps:

- Text Normalization: Preprocess the documents and queries by:
  - Converting text to lowercase.
  - Removing punctuation.
  - Applying stemming (using NLTK or a similar library).
- Write a Python function to:
  - Convert each preprocessed query into a vector based on the terms in the TF-IDF matrix.
  - Calculate cosine similarity for each query against all documents.
  - Rank the documents for each query based on their similarity scores and print the results.