Ex :1

(a) The incidence matrix is a binary matrix that represents the presence or absence of terms in a document corpus. The rows of the matrix represent the terms (or keywords), and the columns represent the documents. If a term appears in a document, the corresponding entry in the matrix is set to 1, otherwise it is set to 0. This representation allows for easy calculation of term frequency (how many times a term appears in a document) and document frequency (how many documents contain a term).

(b) The posting list is a representation of the document corpus as a set of lists, where each list corresponds to a term in the corpus. The lists consist of the document IDs of the documents that contain the term. This representation allows for easy calculation of the inverse document frequency (how rare a term is in the document corpus) and also makes it possible to perform more advanced ranking algorithms such as TF-IDF.

(c) One of the main differences between the incidence matrix and posting list representations is the memory requirements. The incidence matrix representation can require a large amount of memory, especially for a large document corpus with a large number of terms. This is because the matrix is dense, meaning that most entries in the matrix are non-zero. On the other hand, the posting list representation is more memory-efficient, as it only stores the document IDs for each term, rather than a full matrix of values.

(d) The time needed to process a query also differs between the two representations. A query using the incidence matrix representation requires a simple matrix-vector multiplication, which can be performed efficiently using linear algebra techniques. On the other hand, a query using the posting list representation requires iterating through the lists for each term in the query and intersecting the lists to find the documents that contain all of the terms. This process can be more time-consuming than the matrix-vector multiplication, but can be optimized using various techniques such as caching and early stopping. Additionally, the posting list representation can allow for more advanced ranking algorithms, which can result in better results for some queries.

2.The vector model provides a more nuanced representation of the relationships between terms and documents compared to the Boolean model. In a vector model, each document is represented as a vector of term weights, rather than just a binary presence or absence of terms. This allows for a more sophisticated representation of the significance of each term in a document. For example, a term that occurs many times in a document may be considered more significant than a term that occurs only once, and this can be reflected in the weight assigned to that term.

3.In a vector model using a weighting scheme based on the number of term occurrences, the use of the weighting by idf brings additional information about the rarity of a term in the document corpus. The idf (inverse document frequency) weight is calculated as the logarithm of the ratio of the number of documents in the corpus to the number of documents that contain the term. The idf weight therefore reflects how rare a term is in the corpus, with rarer terms having higher idf weights. When combined with the term frequency weight, which reflects the frequency of a term in a document, the idf weight provides a more nuanced representation of the importance of a term in both the document and the corpus as a whole.

Ex 2:

1-There are several methods that can be used to reduce the algorithmic complexity of the document retrieval process:

Indexing: One of the most common methods to reduce the computational cost of document retrieval is to use an indexing structure such as an inverted index. An inverted index is a data structure that maps terms to the documents that contain them, rather than mapping documents to terms. This allows for more efficient searching, as the system only needs to examine the posting list for each term in the query, rather than comparing the query to every document in the corpus.

Document Clustering: Another method is to use document clustering to group similar documents together. This reduces the number of documents that need to be compared to the query, as only the documents in the same cluster as the query need to be considered.

Early Stopping: Another optimization technique is to use early stopping, where the algorithm stops computing scores for documents as soon as a sufficient number of high-scoring documents have been found. This can greatly reduce the computational cost of the algorithm, especially for large document corpora.

Pruning: Another method to reduce the computational cost is to prune the search space by removing terms from the query that are unlikely to appear in relevant documents. This reduces the size of the posting lists that need to be searched, thus reducing the computational cost of the algorithm.

Approximate Matching: Finally, approximate matching techniques can be used to reduce the computational cost of the algorithm. These techniques trade off some accuracy for faster processing by using heuristics to determine the most likely relevant documents, rather than computing scores for all documents in the corpus.

2- There are several methods that aim to guarantee diversity among the documents returned to the user in vector models based on term occurrences:

Clustering: Clustering is a technique that groups similar documents together and can be used to guarantee diversity among the returned documents. The documents can be grouped based on their content, topics, or other relevant features. The k nearest documents can then be selected from different clusters, ensuring that the returned documents are diverse.

Diversity-based ranking: This method involves incorporating a diversity term into the ranking function used to determine the scores between the query and the documents. The diversity term measures the similarity between the already selected documents and the candidate document. The ranking function is then modified to penalize candidate documents that are too similar to the already selected documents.

Diversity promotion: In this method, a diversity promotion term is added to the scores of candidate documents. The diversity promotion term increases the scores of documents that are less similar to the already selected documents, thus promoting diversity among the returned documents.

Re-ranking: Re-ranking is a method that adjusts the scores of the candidate documents after the initial ranking. The adjustment is done based on the diversity between the candidate documents, ensuring that the returned documents are diverse.

In summary, these methods aim to guarantee diversity among the documents returned to the user by either grouping similar documents together, penalizing candidate documents that are too similar to the already selected documents, promoting diversity, or adjusting the scores of the candidate documents. By incorporating diversity into the scoring or ranking process, these methods help ensure that the returned documents are diverse and representative of the corpus.

Ex 3:

Briefly remind the principles of latent semantic indexing. How is it implemented ? What are the guiding ideas ?

Latent Semantic Indexing (LSI) is a technique for information retrieval that is based on the idea that documents that are semantically similar should also have similar patterns of term usage. It is implemented by creating a low-dimensional semantic representation of the documents in the corpus, where the representation is learned from the co-occurrence patterns of terms within documents.

The guiding idea behind LSI is that by reducing the dimensionality of the document representations, it is possible to capture the underlying meaning and relationships between documents, even if they do not share many terms. This is accomplished by constructing a term-document matrix, where each document is represented as a vector of term weights, and then using singular value decomposition (SVD) to factor the matrix into three matrices. The left singular vectors in the low-rank approximation of the term-document matrix capture the semantic concepts represented in the documents, while the right singular vectors capture the relationships between terms.

Once the semantic representation of the documents has been computed, it is possible to perform information retrieval tasks such as query-document similarity computation and document ranking, by using the same techniques used in vector space models. In practice, LSI can be used to improve the effectiveness of information retrieval systems, especially for queries that contain multiple related concepts, by better capturing the relationships between documents and queries.

A stemmer is a tool used in natural language processing for reducing words to their base or root form, known as the stem. The stem is the core of the word, to which affixes can be added to create different variations of the word. The goal of stemming is to reduce words to their core structure, so that they can be easily matched with related words in a text corpus. This is particularly useful in information retrieval, where it helps to reduce the dimensionality of the data and improve the recall of text-based searches. The stemmer typically removes suffixes and morphological endings, such as -ing, -ed, or -s, to produce a stemmed form of the word that can be matched with similar words in a corpus.

The influence of stemming on the spatial complexity of the IR process can depend on the specific implementation and the size of the corpus. In general, stemming can reduce the size of the index by grouping together words with similar roots, which can lead to reduced memory requirements. However, the processing cost of stemming can also add additional computational overhead.

The document collection is:  
d1 ={t2 t4 t3 t4 t1 t2}  
d2={t1 t3 t2}  
d3={t4 t5 t2 t2 t4}

D4={t5 t1 t2 t1 t3 t2}

D5={t2 t5 t1 t4 t1}

binary model :  
give me for each term t the posting list in the order of increasing document indices

give the vectorial representation of all the docs according to the tf weighting scheme

let the following results be obtained for queries q1 and q2:

q1: d1,d2,d3,d4,d5,d6,d7,d8,d9,d10

q2 : d1,d2,d3,d4,d5,d6,d7,d8,d9,d10

and the relevant documents for q1 are: d1,d3,d5,d6

and the relevant documents for q2 are:d1,d3,d5

whereas for q1 we have 4 relevant documents and for q2 we have 5 , so 2 relevant document have not been found for q2.

1. For the recall values k/10 (for k in 1,…,10) give the value of the averaged interpolated precision for the two queries. Detail the intermediate steps