Q1.

Given the document collection:

D1: Shipment of gold damaged in a fire

D2: Delivery of silver arrived in a silver truck

D3: Shipment of gold arrived in a truck

And the query:

Q: gold silver truck

Steps to be performed to rank the document:

1. **Document Pre-processing:**
   Stop word removal process can be used to pre-process the document. It is a removal of most frequent words from the document which add very little to the meaning or semantics of the document. Thus, in the given document frequency of the word 'of', 'in', 'a' is 3 which is equal to the number of documents. Hence, we can remove these words as they are not relevant.
   Another pre-processing step can be converting all words to **lower case.** Thus, the words 'Shipment', 'Delivery' will be converted to 'shipment', 'delivery'. This step is important as the frequency of words calculated should be case independent. Hence, we perform this step to make sure we get the correct frequency of the term.
2. **Model Selection:**
   To rank the documents, we need to choose a model. In this case, I will use **Vector Space Model**. This model is an improvement over Boolean model which can represent term only in form of 0's and 1's. Here, terms can have non-binary weights in both query and documents. Also, query and documents can be represented as n-dimensional vectors.
3. **Representation of Document and Query:**
   Each term in a document and query can be represented in form of **weighing schemes**. We need a means to calculate the term weights in the document and query vector. tf-idf is the most popular weighing scheme. **Term frequency (tf)** is a factor which tells how well a term describes the document. The more the frequency of the term, the better it is at describing a document. **Inverse Document Frequency (idf)** on the other hand is a factor of the term that occurs frequently across all the documents and hence, it is not significant in distinguishing one document from another.
   Thus, we calculate these measures in the given document.
   tf= frequency of term i in document j
   df= number of documents a term i appears in
   idf= how discriminating a term i is which can be calculated as
   $log(\frac{N}{N_i})$ where N= total number of documents, N(i)= df
   We then calculate the weight of each term using below formula:
   $w_{i,j} = f_{i,j} \times log(\frac{N}{N_i})$ where f (i, j) is term frequency (tf) and log (N/N(i)) is idf.

The resultant table shown below represents the document and query as vectors. Weight calculated acts as coordinates in vector space.

Terms	Counts, tf(i)							Weights w(i)= tf(i) * idf(i)			
	Q	D1	D2	D3	df(i)	D/df(i)	idf(i)= log(D/df(i))	Q	D1	D2	D3
shipment	0	1	0	1	2	1.5	0.1761	0	0.1761	0	0.1761
gold	1	1	0	1	2	1.5	0.1761	0.1761	0.1761	0	0.1761
damaged	0	1	0	0	1	3	0.4771	0	0.4771	0	0
fire	0	1	0	0	1	3	0.4771	0	0.4771	0	0
delivery	0	0	1	0	1	3	0.4771	0	0	0.4771	0
silver	1	0	2	0	1	3	0.4771	0.4771	0	0.9542	0
arrived	0	0	1	1	2	1.5	0.1761	0	0	0.1761	0.1761
truck	1	0	1	1	2	1.5	0.1761	0.1761	0	0.1761	0.1761

4. **Comparison:**

Similarity measure is the cosine angle between two vectors. We must calculate similarity between document and query using term weights by the below formula:

$$sim(q,d) = \frac{\sum_{1=1}^{N}(w_{i,q} \times w_{i,d})}{\sqrt{\sum_{1=1}^{N}(w_{i,q})^2} \times \sqrt{\sum_{i=1}^{N}(w_{i,d})^2}}$$

|Q| = square root of (0.1761 ^2) + (0.4771 ^2) + (0.1761 ^2) = 0.5382
|D1| = square root of (0.1761^2) + (0.1761^2) + (0.4771^2) +(0.4771^2) = 0.7192
|D2| = square root of (0.4771^2) + (0.9542^2) + (0.1761^2) + (0.1761^2) = 1.0955
|D3| = square root of (0.1761^2) + (0.1761^2) + (0.1761^2) + (0.1761^2) = 0.3522

Q*D1 = (0*0.1761) + (0.1761*0.1761) + (0*0.4771) + (0*0.4771) + (0*0) + (0.4771*0) + (0*0) + (0.1761*0) = 0.0310
sim (Q, D1) = 0.0310 / (0.5382 * 0.7192) = 0.0801

Q*D2 = (0*0) + (0.1761*0) + (0*0) + (0*0) + (0*0.4771) + (0.4771*0.9542) + (0*0.1761) + (0.1761*0.1761) = 0.4863
sim (Q, D2) = 0.4863 / (0.5382 * 1.0955) = 0.8248

Q*D3 = (0*0.1761) + (0.1761*0.1761) + (0*0) + (0*0) + (0*0) + (0.4771*0) + (0*0.1761) + (0.1761*0.1761) = 0.0620
sim (Q, D3) = 0.0620 / (0.5382 * 0.3522) = 0.3271

5. **Ranked List:**

Based on similarity obtained in previous step the rank of document will **D2, D3, and D1**.

Assumptions:

1. Terms are considered independent of each other.
2. |Q|, |D1|, |D2|, |D3| are normalisation factor in calculation of weight.
3. I have removed words like ‘a’, ‘of’, ‘in’ as they had maximum frequency across all the documents, hence, I considered them to be non-relevant.

4. We'll assume that the more times a document contains a term, the more likely it is to be *about* that term.

Q2.

1. D1 = Shipment of gold damaged in a fire. Fire.
2. D1 = Shipment of gold damaged in a fire. Fire. Fire.
3. D1 = Shipment of gold damaged in a fire. Gold.
4. D1 = Shipment of gold damaged in a fire. Gold. Gold.

These can be explained by axiomatic approaches.

1. D1 = Shipment of gold damaged in a fire. Fire.
   Here a new term 'Fire' is added. If we ignore the uppercase of the first letter, then it is equivalent to 'fire'. Hence, the term frequency of fire would increase. 'fire' is a non-query term. According to **Constraint 2** of axiomatic approach,
   document (1) = document U (union) {term} | term does not belong to query then,
   similarity (document (1), query) < similarity (document, query)
   It states that if a term is added to a document, and it is a non-query term then the score/similarity of the modified document will always be less than the similarity of the original document. **Thus, the similarity of D1 sim (Q, D1) will decrease in the given case.** This can also be verified by the fact that the new similarity sim (Q, D1) = 0.0526 which is less than 0.0801 similarity of the original document.

2. D1 = Shipment of gold damaged in a fire. Fire. Fire.
   Here the term 'fire' is occurring successively in the document which is a non-query term. According to the **Constraint 3 and 4** of axiomatic approach, adding successive occurrence of a term to a document which is a non-query term must decrease the score of the document in a sub-linear way i.e.,
   similarity (document (1), query) – similarity (document, query) > similarity (document (2), query) – similarity (document (1), query)
   Thus, the similarity of this document will **further decrease** from the previous case as 'fire' is a non-query term. This can be verified by the fact that similarity sim (Q, D1) for this case is 0.0377. Thus, 0.0801 – 0.0526 > 0.0526 – 0.0377.

3. D1 = Shipment of gold damaged in a fire. Gold.
   A new term 'gold' is added here which is a query term. According to the Constraint 1 of axiomatic approaches, adding a query term to a document must always increase the score of the document.
   document (1) = document U (union) {term} | term belongs to query then,
   similarity (document (1), query) > similarity (document, query)
   Thus, the similarity in this case for D1 will increase. This can be verified by the fact that the new similarity sim (Q, D1) = 0.1475 which is greater than 0.0801 similarity of the original document.

4. D1 = Shipment of gold damaged in a fire. Gold. Gold.
   Successive term 'gold' is added here which is a query term. According to **Constraint 3** of axiomatic approach, adding successive occurrences of a term to a document must increase the score of the document in a sub-linear way.

   document (1) = document U (union) {term} | term belongs to query,

   document (2) = document (1) U (union) {term} | term belongs to query

similarity (document (1), query) – similarity (document, query) > similarity (document (2), query) – similarity (document (1), query)

Thus, the similarity of the document will further increase i.e., 0.2258 in this case.

Q3.

Assuming that your document collection consists of all the scientific articles published in the Communications of the ACM (www.acm.org/dl), identify two other sources of evidence (features or sets of features) one could consider and suggest a weighting scheme that incorporates these features.
Your answer should define the evidence/feature, your reason for including it, and a means to include it in the weighting scheme.

Answer: Two other sources of evidence:

1. **Category Information across all the collection**:
   We can use category information of any article to determine whether it is similar to our query or not. Category for a scientific article can be technology, environment, work, computer etc. We not only find the category across all the documents but across all the collection, in this case scientific article collection. Thus, it acts as a global factor in weighing scheme.

   Reasons for including it:
   Category information associated with the document can serve as a guidance in determining which terms are more informative than others. We can find such meta information along with documents in collections like scientific articles. We assume that the semantic of a document can be represented by the set of categories that it belongs to. Thus, by measuring the similarity in category labels assigned to two documents, we will be able to tell content wise how similar they are.

2. **Document length and reasons for including it:**
   The length of the document can play a major role in determining the similarity of the document. For example, a term 'computer' occurs twice in a short document and a long document. The relevance of the term is more in a short document than in the long document. It is highly likely that the short document is about computers, however, it is not necessary that the long document is about computers as well. Thus, we should reward matches in short document and penalize matches in long document.

   Weighing Scheme:
   BM25 is a modern weighing scheme which can include both features. BM (Best Matching) 25 originally known as Okapi BM25 is a modern ranking function used to find the relevance of a document to a given search query. The modern global weighing scheme has following components:

$$sim(q,d) = \sum_{t \in q \cup d} (ntf(D) \times gw_t(C) \times qw_t(Q))$$

   Where ntf(D) is the normalized term frequency in a document,
   gw(C) is the global weight of a term across a collection,
   qw(Q) is the query weight of a term in Q.

The above global weighing scheme is represented well in BM25 as well.

$$BM25(Q,D) = \sum_{t \in Q \cap D} \left(\frac{tf_t^D \cdot \log\left(\frac{N - df_t + 0.5}{df_t + 0.5}\right) \cdot tf_t^Q}{tf_t^D + k_1 \cdot ((1-b) + b \cdot \frac{dl}{dl_{avg}})} \right)$$

Here, tf (t, D) is the term frequency in a document,
log (N - df + 0.5/ df + 0.5) is similar to idf factor which corresponds to global weight across collection,
tf (t, Q) is the query weight.

- In order to **include document length** in the weighing scheme, we have to decide a frame of reference. We can use the corpus itself as the frame of reference. Thus, a short document is the one that is shorter than average for the corpus and long document is longer than average. This can be achieved by the following formula **dl/dl(average)** which is used in the denominator of BM25.
- If we want to **include category** in the BM25 formula, we can replace the idf (inverse document frequency) factor with the icf (inverse category frequency) to identify the importance of term. For each term 't', we compute the inverse category frequency i.e., **icf(t) = log m – log mc(t),** where m is the number of categories and mc(t) is the number of categories that contain the term t. Now this icf value can be used in BM25 formula, thus, idf value of log (N - df + 0.5/ df + 0.5) will be replaced with icf(t). This approach is based on the assumption that a word tends to be less informative when it appears across a large number of categories. We hypothesize that category frequency could be a better indicator for term importance than the document frequency. In an extreme case where a document is repeated thousand of times in a collection. A document frequency will assign low weights to any words in a document while category frequency-based approach will not.

References:

1. Rong Jin, Joyce Y. Chai and Luo Si (2005). "Learn to weight terms in Information Retrieval using Category Information".
2. https://kmwllc.com/index.php/2020/03/20/understanding-tf-idf-and-bm-25/