Problems on Boolean query model using incidence matrix

1.Consider the following collection of documents and set of index terms,

D1: Frodo stabbed the orc with the red sword.

D2: Frodo and Sam used the blue lamp to locate orcs.

D3: Sam killed many orcs in Mordor with the blue sword.

K={Frodo, Sam, blue, sword, orc, Mordor}

For the **given Boolean query find the relevant documents using incidence matrix** or inverted file.

Query: (Frodo AND orc AND sword) OR (Frodo AND blue)

Solution:

Construction of Term-document incidence matrix:

Term	D1	D2	D3
Frodo	1	1	0
Red	1	0	0
Sam	0	1	1
Blue	0	1	1
Sword	1	0	1
Orc	1	1	1
Mordor	0	0	1

Query : (Frodo AND orc AND sword) OR (Frodo AND blue)

Term	D1	D2	D3
Frodo	1	1	0
Sword	1	0	1
Orc	1	1	1
Blue	0	1	1

Frodo = [1,1,0]

Orc = [1,1,1]

Sword = [1,0,1]

Blue= [0,1,1]

(Frodo AND orc AND sword) OR (Frodo AND blue)

([1, 1, 0] AND [1,1,1] AND [1,0,1]) OR ([1,1,0] AND [0,1,1])

= [1,0,0] OR [0,1,0] = [1,1.0]

D1, D2 are relevant document for the query.

2.Consider the following corpus:

D1: "The quick brown fox jumps over the lazy dog"

D2: "The brown dog jumps over the quick fox"

D3: "The lazy brown dog jumps over the quick fox"

For the given Boolean query find the relevant documents using incidence matrix or inverted file.

Query: Fox AND lazy

Solution:

Index terms considered after removing stop words like the, over –

K= quick, brown, fox, jumps, lazy dog

Construction of Term-document incidence matrix:

Term	D1	D2	D3
Quick	1	1	1
Brown	1	0	1
Fox	1	1	1
Jumps	1	1	1
Lazy	1	0	1
Dog	1	1	1

Query : **Fox AND lazy**

Term	D1	D2	D3
Fox	1	1	1
Lazy	1	0	1

Fox : [1,1,1]

Lazy: [1,0,1]

Fox AND lazy = [1,1,1] AND [1,0,1] = [1,0,1]

D1, D3 are relevant document for the query.

3. Consider the following collection of documents and set of index terms,

D1: Mickey Mouse is the mascot of The Walt Disney Company. Mickey generally appears alongside his girlfriend Minnie Mouse, his pet dog Pluto, his friends Donald Duck and Goofy,

D2: Pluto is a cartoon character created by The Walt Disney Company. Pluto is Mickey Mouse's pet dog.

D3: Goofy is a cartoon character created by The Walt Disney Company. He is a tall, anthropomorphic dog. Goofy is a close friend of Mickey Mouse and Donald Duck.

K={Mickey, Mouse, Walt, Disney, Minnie, Pluto, Dog, Donald, Duck, Goofy}

For the given Boolean query find the relevant documents using incidence matrix or inverted file.

Query: Mickey AND (Dog OR Donald)

Solution:

Construction of Term-document incidence matrix:

Term	D1	D2	D3
Mickey	1	1	1
Mouse	1	1	1
Walt	1	1	1
Disney	1	1	1
Minnie	1	0	0
Pluto	1	1	0
Dog	1	1	1
Donald	1	0	1
Duck	1	0	1
goofy	1	0	1

Query : Mickey AND (Dog OR Donald)

Term	D1	D2	D3
Mickey	1	1	1
Dog	1	1	1
Donald	1	0	1

[1,1,1] AND [[1,1,1] OR [1,0,1]]

[1,1,1] AND [1,1,1] = [1,1,1]

D1,D2,D3 are relevant documents.

Problems on vector model using term frequency-inverse document frequency (TF-IDF) weighting

1. Consider the following documents

D1: "The quick brown fox jumps over the lazy dog"

D2: "The brown dog jumps over fox"

D3: "The lazy dog jumps over the quick fox"

Using **term frequency-inverse document frequency (TF-IDF) weighting,** construct a document-term matrix for this corpus. Now, suppose a user enters the query **"lazy dog".** Using cosine similarity, find the ranking of documents based on their relevance to the query.

Solution:

Step 1: (If index term not given consider your own set by removing stopwords).

To construct the document-term matrix using TF-IDF weighting, we first create a vocabulary of all unique terms in the corpus. In this case, the vocabulary is: **{The, quick, brown, fox, jumps, over, lazy, dog}**

Step 2: Calculate frequency count and IDF

$$idf_i = \log \frac{N}{n_i}$$

Document – Term frequency count									
	The	quick	brown	fox	jumps	Over	Lazy	dog	MAX
D1	2	1	1	1	1	1	1	1	2
D2	1	0	1	1	1	1	0	1	1
D3	2	1	0	1	1	1	1	1	2
IDF	0	0.176	0.176	0	0	0	0.176	0	

Step3 : Calculate term weight for document

$$f_{i,j} = \frac{freq_{i,j}}{\max_l freq_{l,j}} \qquad w_{i,j} = freq_{i,j} \times idf_i$$

Weight of index terms in document [wi,j = freqi,j X idf]								
	The	quick	brown	fox	jumps	Over	Lazy	dog
D1	0	0.088	0.088	0	0	0	0.088	0
D2	0	0	0.176	0	0	0	0	0
D3	0	0.088	0	0	0	0	0.088	0

Step 3: calculate term weight for query.

Query - **"lazy dog"**

$$w_{i,q} = (0.5 + \frac{0.5\, freq_{i,q}}{\max_l freq_{l,q}}) \times \log \frac{N}{n_i}$$

	The	**quick**	**brown**	**fox**	**jumps**	**Over**	**Lazy**	**dog**	**MAX**
	0	0	0	0	0	0	1	1	1
Freq	0.5	0.5	0.5	0.5	0.5	0.5	1	1	
IDF	0	0.176	0.176	0	0	0	0.176	0	
wiq	0	0.088	0.088	0	0	0	0.176	0	

$$sim(d_j, q) = \frac{\vec{d}_j \cdot \vec{q}}{|\vec{d}_j| \times |\vec{q}|}$$

$$= \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}}$$

Sim(d1,q)= 0.485

Sim(d2,q)= 0.205

Sim(d3,q)= 0.435

Based on similarity, documents can be ranked as d1,d3,d2

2. Consider the set of documents,

D1= The Ganga is one of the world's great rivers. Its wide valley stretches across northern India and Bangladesh from the Himalayas to the Bay of Bengal.

D2= The Kaveri River, also spelled Cauvery, sacred river of southern India, is famous as the Ganga of the South. The Kaveri River ultimately drains into the Bay of Bengal.

D3 = The Godavari is India's second longest river after the Ganga River and drains into the Bay of Bengal

K= {Ganga, River, India, Bay, Bengal, Kaveri, Godavari}

Using **term frequency-inverse document frequency (TF-IDF) weighting**, construct a document-term matrix for this corpus. Now, suppose a user enters the query **"River Kaveri India"**. Using cosine similarity, find the ranking of documents based on their relevance to the query.

Step 1: Calculate frequency count and IDF

$$idf_i = \log \frac{N}{n_i}$$

K= Ganga, River, India, Bay, Bengal, Kaveri, Godavari

Document – Term frequency count								
	Ganga	River	India	Bay	Bengal	Kaveri	Godavari	MAX
D1	1	1	1	1	1	0	0	1
D2	1	3	1	1	1	2	0	3
D3	1	1	1	1	1	0	1	1
IDF	0	0	0	0	0	0.477	0.477	-

Step2 : Calculate term weight for document

$$f_{i,j} = \frac{freq_{i,j}}{\max_l freq_{l,j}} \qquad w_{i,j} = freq_{i,j} \times idf_i$$

Weight of index terms in document [wi,j = freqi,j X idf]							
	Ganga	River	India	Bay	Bengal	Kaveri	Godavari
D1	0	0	0	0	0	0	0
D2	0	0	0	0	0	0.318	0
D3	0	0	0	0	0	0	0.477

Step 3: calculate term weight for query.

Query - **River Kaveri India**

$$w_{i,q} = (0.5 + \frac{0.5\, freq_{i,q}}{\max_l freq_{l,q}}) \times \log \frac{N}{n_i}$$

Weight of index terms in Query [wi,q = freq X idf]								
	Ganga	River	India	Bay	Bengal	Kaveri	Godavari	Max
Q	0	1	1	0	0	1	0	1
Freq	0.5	1	1	0.5	0.5	1	0.5	
IDF	0	0	0	0	0	0.477	0.477	
Wiq	0	0	0	0	0	0.477	0.238	

$$sim(d_j, q) = \frac{\vec{d}_j \cdot \vec{q}}{|\vec{d}_j| \times |\vec{q}|}$$

$$= \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}}$$

Sim (d1, q) = 0

Sim (d2, q) = 0.1516/0.1695=0.8944

Sim (d3, q) =0.1135/0.2542=0.4464

Based on similarity, documents can be ranked as d2, d3,d1

3. Consider the following collection of documents and set of index terms,

D1: Mickey Mouse is the mascot of The Walt Disney Company. Mickey generally appears alongside his girlfriend Minnie Mouse, his pet dog Pluto, his friends Donald Duck and Goofy,

D2: Pluto is a cartoon character created by The Walt Disney Company. Pluto is Mickey Mouse's pet dog.

D3: Goofy is a cartoon character created by The Walt Disney Company. He is a tall, anthropomorphic dog. Goofy is a close friend of Mickey Mouse and Donald Duck.

K={Mickey, Mouse, Walt, Disney, Minnie, Pluto, Dog, Donald, Duck, Goofy}

Using **term frequency-inverse document frequency (TF-IDF) weighting,** construct a document-term matrix for this corpus. Now, suppose a user enters the query **"Mickey Donald Dog "**. Using cosine similarity, find the ranking of documents based on their relevance to the query.

Step 1: Calculate frequency count and IDF

$$idf_i = \log \frac{N}{n_i}$$

Document – Term frequency count											
	Mickey	Mouse	Walt	Disney	Minnie	Pluto	Dog	Donald	duck	goofy	MAX
D1	2	2	1	1	1	1	1	1	1	1	2
D2	1	1	1	1	0	2	1	0	0	0	2
D3	1	1	1	1	0	0	1	1	1	2	2
IDF	0	0	0	0	0.477	0.176	0	0.176	0.176	0.176	

Step2 : Calculate term weight for document

$$f_{i,j} = \frac{freq_{i,j}}{\max_l freq_{l,j}} \qquad w_{i,j} = freq_{i,j} \times idf_i$$

Weight of index terms in document [wi,j = freqi,j X idf]									
	Mickey	Mouse	Walt	Disney	Minnie	Pluto	Dog	Donald	duck
D1	0	0	0	0	0.238	0.088	0	0.088	0.088
D2	0	0	0	0	0	0.176	0	0	0
D3	0	0	0	0	0	0	0	0.088	0.088

Step 3: calculate term weight for query.

Query - **"Mickey Donald Dog "**

$$w_{i,q} = (0.5 + \frac{0.5\, freq_{i,q}}{\max_l freq_{l,q}}) \times \log \frac{N}{n_i}$$

Weight of index terms in Query [wi,q = freq X idf]											
	Mickey	Mouse	Walt	Disney	Minnie	Pluto	Dog	Donald	duck	goofy	MAX
Q	1	0	0	0	0	0	1	1	0	0	1
freq	1	0.5	0.5	0.5	0.5	0.5	1	1	0.5	0.5	
IDF	0	0	0	0	0.477	0.176	0	0.176	0.176	0.176	
Wi,q	0	0	0	0	0.238	0.088	0	0.176	0.088	0.088	

$$sim(d_j, q) = \frac{\vec{d}_j \cdot \vec{q}}{|\vec{d}_j| \times |\vec{q}|}$$

$$= \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}}$$

Sim(d1,q)=0.0956/(0.2964*0.333) =0.9685
Sim(d2,q)=0.0154/(0.176*0.333)=0.2627
Sim(d3,q)= 0.03872/(0.2155*0.333)=0.5395
Based on similarity, documents can be ranked as d1,d3,d2