Name: Sai Anish Garapati
UIN: 650208577

Midterm-1:

Exercise 1:

1. Term Document matrix:

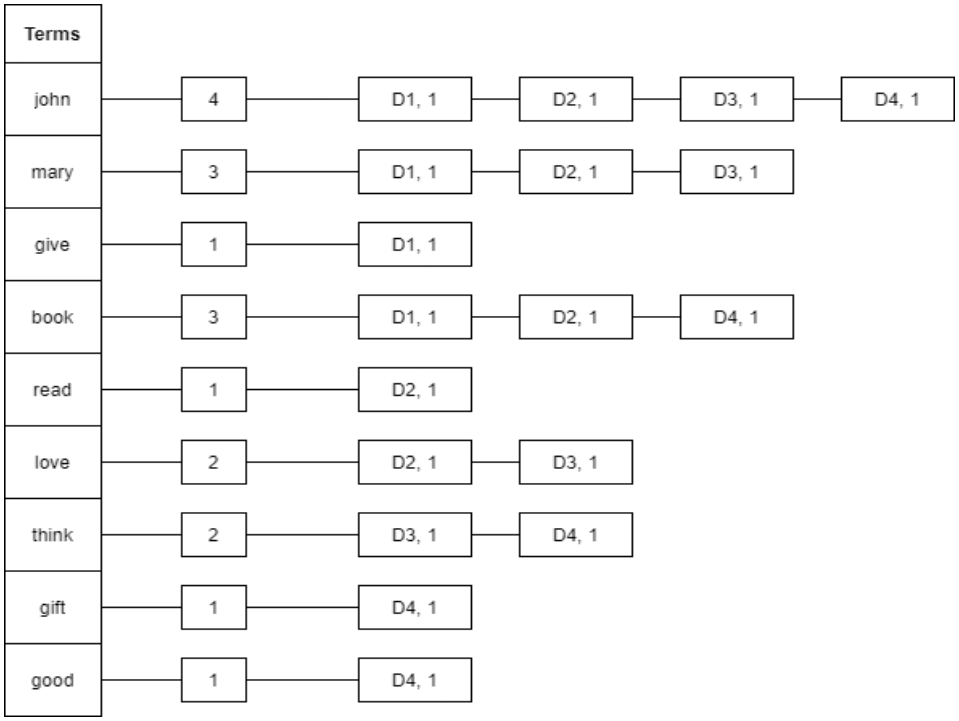
Term	Doc1	Doc2	Doc3	Doc4
john	1	1	1	1
mary	1	1	1	0
give	1	0	0	0
book	1	1	0	1
read	0	1	0	0
love	0	1	1	0
think	0	0	1	1
gift	0	0	0	1
good	0	0	0	1

2. Docs returns for the following queries

- (1, 2, 3, 4 AND 4) OR (3, 4) = (4) OR (3, 4) = **Docs 3, 4**
- (1, 2, 3, 4 AND 1, 2, 3 AND 1, 2) = **Docs 1, 2**
- (1, 2, 4) AND (2 OR 1, 2, 3, 4) = (1, 2, 4) AND (1, 2, 3, 4) = **Docs 1, 2, 4**

Exercise 2:

1. Stemming the corpus would have a negative impact on precision for some queries. If the documents contain many terms that would be stemmed to the same word, but a word before stemming may be a commonly occurring term with slightly different meaning. Then we would lose considerable precision for queries including that term, as documents containing all the derivatives from the stemmed word would be returned.
2. Inverted index in the form Term-> Document_frequency-> <doc number, term frequency> lists



3. Initialize a hashtable Rank = {} which will use docs D1, D2, D3, D4 as the key entries and update the values as we parse through the query. Parsing through the query, we get:
   - The first word “love” in the inverted index is present in docs D2, D3. Initializing entries D2, D3 in the hashtable with the weight information we get:
     $Rank[D2] -> w_{2,love} * w_{q,love}$
     $Rank[D3] -> w_{3,love} * w_{q,love}$
   - The second word “mary” in the inverted index is present in docs D1, D2, D3. Initializing entry D1 and updating the values for entries D2, D3 with the weight information we get:
     $Rank[D1] -> w_{1,mary} * w_{q,mary}$
     $Rank[D2] -> w_{2,love} * w_{q,love} + w_{2,mary} * w_{q,mary}$
     $Rank[D3] -> w_{3,love} * w_{q,love} + w_{3,mary} * w_{q,mary}$

   Where, $w_{i,word} = tf_{i,word} * \log_2(\frac{N}{df_{word}}) = \frac{f_{i,word}}{max\{fi\}} * \log_2(\frac{N}{df_{word}})$ where ‘i’ can be a document id or query.

   Using tfs and dfs from inverted index table and using the above formula we get:
   $Rank[D1] = 0.1722,\ Rank[D2] = 1.1722,\ Rank[D3] = 1.1722$
   These final values are then normalized by product of length of document and length of query.
   $len[Di] = \sqrt{\sum_{word\ \epsilon\ Di} (w_{i,word})^2}$
   Using , we get:
   $len[D1] = 2.084,\ len[D2] = 2.25,\ len[D3] = 1.42,\ len[q] = 1.082$
   Then we get $Rank[D1] = 0.076, Rank[D2] = 0.48, Rank[D3] = 0.76, Rank[D4] = 0$ (weights=0)
   Ranking of the documents based on cosine similarity is $D3 > D2 > D1 > D4$.

Exercise 3:

Probability estimates for each term in the documents:

	language	vision	human	robot	interaction	speech	natural
D1	3/7	1/7	1/7	1/7	1/7	0	0
D2	1/2	0	0	0	0	1/2	0
D3	1/5	1/5	0	0	0	2/5	1/5

Probability estimates for each term in the collection:

	language	vision	human	robot	interaction	speech	natural
Collection	5/14	2/14	1/14	1/14	1/14	3/14	1/14

Calculating the model probabilities for the query for each document we get:

P(Q | D1) = [0.8 * 2/14 + 0.2 * 1/7] * [0.8 * 5/14 + 0.2 * 3/7] = 0.053

P(Q | D2) = [0.8 * 2/14 + 0.2 * 0] * [0.8 * 5/14 + 0.2 * 1/2] = 0.044

p(Q | D3) = [0.8 * 2/14 + 0.2 * 1/5] * [0.8 * 5/14 + 0.2 * 1/5] = 0.0502

Ranking of the Documents based on above probabilities is $D1 > D3 > D2$

Exercise 4:

Representing documents in binary vectors based on the vocabulary we get:

$Doc\ 1 : < 1, 1, 1, 1, 0, 0, 0, 0, 0 >$
$Doc\ 2 : < 1, 1, 0, 1, 1, 1, 0, 0, 0 >$
$Doc\ 3 : < 1, 1, 0, 0, 0, 1, 1, 0, 0 >$
$Doc\ 4 : < 1, 0, 0, 1, 0, 0, 1, 1, 1 >$

Representing query in binary vector form based on the vocabulary we get:

$Query(q_0) : < 0, 1, 0, 0, 0, 1, 0, 0, 0 >$

$$\frac{1}{|D_r|} \sum_{d_j\ \epsilon\ D_r} \vec{d_j} = \frac{1}{1} * (< 1, 1, 0, 1, 1, 1, 0, 0, 0 >)$$
$$= < 1, 1, 0, 1, 1, 1, 0, 0, 0 >$$

$$\frac{1}{|D_{nr}|} \sum_{d_j\ \epsilon\ D_{nr}} \vec{d_j} = \frac{1}{3} * (< 1, 1, 1, 1, 0, 0, 0, 0, 0 > + < 1, 1, 0, 0, 0, 1, 1, 0, 0 >$$
$$+ < 1, 0, 0, 1, 0, 0, 1, 1, 1 >)$$
$$= \frac{1}{3}(< 3, 2, 1, 2, 0, 1, 2, 1, 1 >)$$

$$\vec{q_m} = 1 * < 0, 1, 0, 0, 0, 1, 0, 0, 0 > + 1 * < 1, 1, 0, 1, 1, 1, 0, 0, 0 > - 1 *$$
$$\frac{1}{3} * (< 3, 2, 1, 2, 0, 1, 2, 1, 1 >)$$

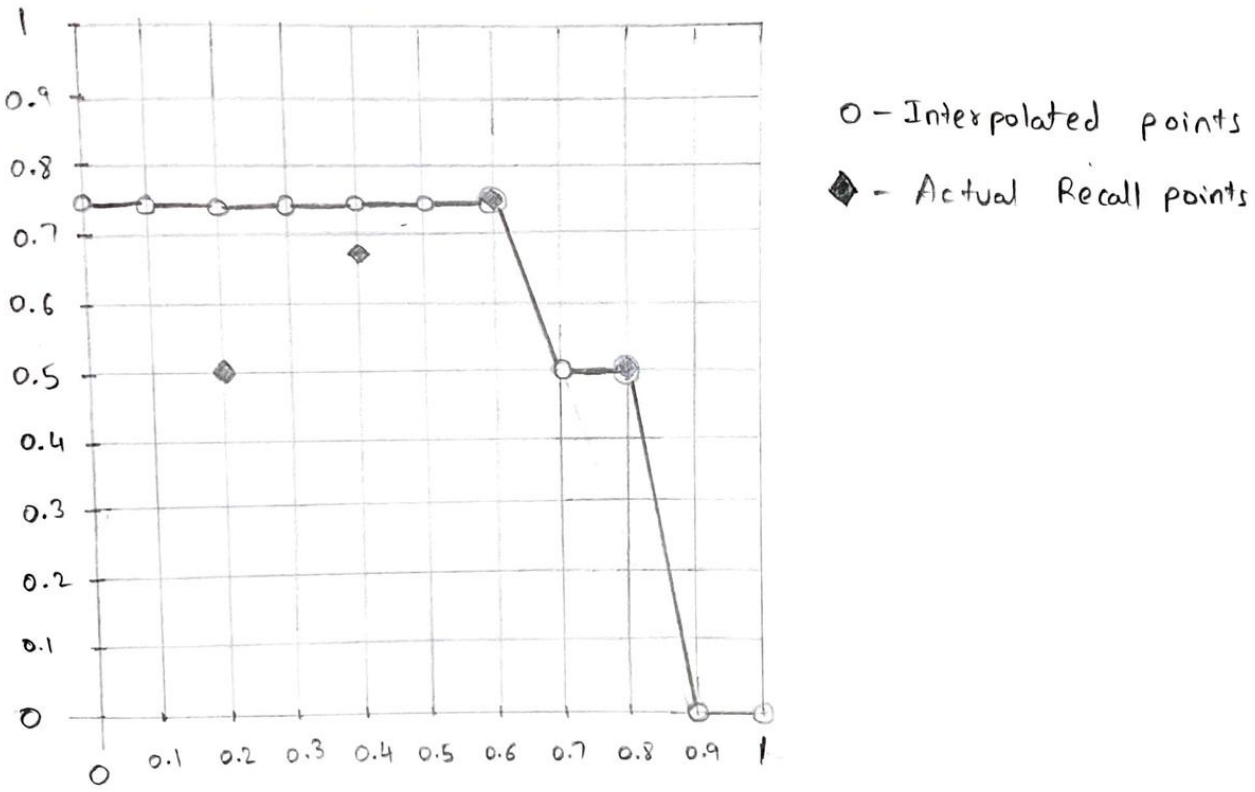
$$\vec{q_m} = < 0, \frac{4}{3}, \frac{-1}{3}, \frac{1}{3}, 1, \frac{5}{3}, \frac{-2}{3}, \frac{-1}{3}, \frac{-1}{3} >$$

Ignoring the negative term weights, we get the updated query vector as:

$$\vec{q_m} = < 0, \frac{4}{3}, 0, \frac{1}{3}, 1, \frac{5}{3}, 0, 0, 0 > = < 0,\ 1.33,\ 0,\ 0.33,\ 1,\ 1.67,\ 0,\ 0,\ 0 >$$

Exercise 5:

1. $Prec(r) = \max_{r'>=r} Prec(r')$



2. - Precision at first recall point(0.2) is 0.5. For this 2nd document must be relevant.
   - Precision is seen to be increasing for the next two recall points (0.4 and 0.6). So the next documents 3 and 4 are relevant to match the precision.
   - Precision at the last recall point(0.8) is 0.5. For this to happen there must be equal non-relevant and relevant documents. To match this document 8 is relevant.

D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
	R	R	R				R		
	P = 0.5, R = 0.2	P=0.67, R = 0.4	P=0.75, R = 0.6				P = 0.5, R = 0.8		

3. Total number of relevant documents = 5
   R-precision value at the 5th document in the retrieved ranked documents from the table in the above question = 3/5 = 0.6