Question-1

Pre_Processing:

1. Loaded dataset
2. Removed Numbers
3. Converted data to lowercase
4. Removed Punctuations,stopwords , 2 length words
5. Lemmatized text

Approach :

Part (a):

1. Written method to find Union and Intersection for document and Query.
2. Written method to compute Jaccard coefficient.
3. Written method to get the top 5 documents on the basis of Jaccard coefficient.

Part (b):

1. Preprocessed data , generated postings list and frequency count list
2. Further computed term frequency for all five variant scheme
    a. Binary : 0 , 1
    b. Raw Count : f(t,d)
    c. Term frequency : f(t,d)/Pf(t‘, d)
    d. Log normalization : log(1+f(t,d))
    e. Double normalization : 0.5+0.5*(f(t,d)/ max(f(t‘,d))

3. Created a dictionary to store term frequencies in all docs.
4. Further computed matrices for all five variant scheme

Question 2 :

Pre-processing

1. Loaded the dataset using Pandas library
2. Retrieved docs with qid:4 using groupby() function
3. Converted the string values to float

Methodology:

1. To find the max_dcg value, we have sorted the relevance_judgement_score in descending order as it will give max_dcg.

2. To find the total number of documents with max_dcg, we are counting the frequency of each relevance_judgement_score and find all the possible permutations.

3. To find the NDCG value, we are using the below formula and calculating DCG and Ideal DCG values. (NDCG = DCG/IDCG)

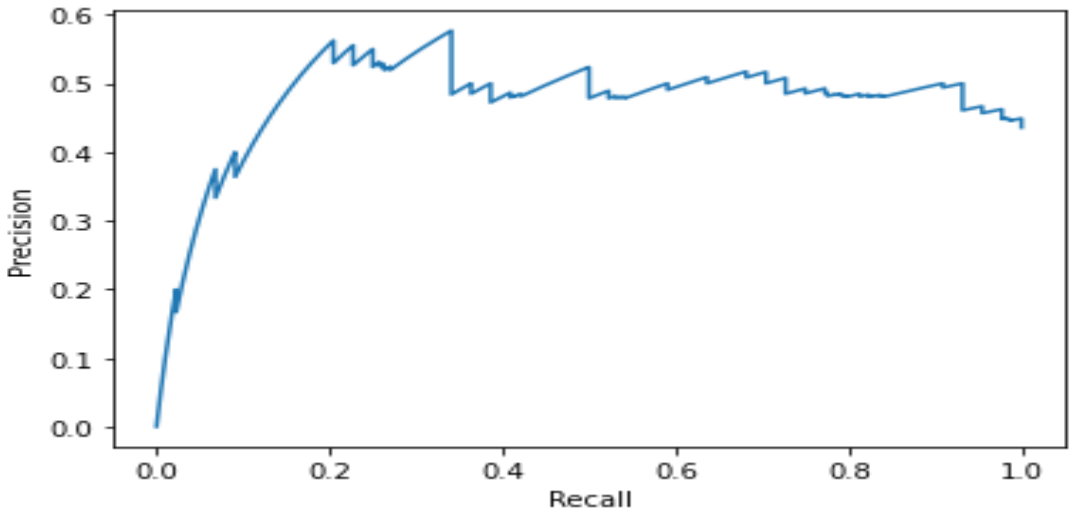
$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{log(1+i)}$$

4. To plot the Precision Recall curve, we are finding the precision and recall values for each query using the actual relevance values and Feature_75 values which are normalized between 0-1.

Assumptions:

No assumptions

Precision-Recall Curve



Question-3:

Pre_Processing:

6. Loaded dataset
7. Removed Numbers
8. Converted data to lowercase
9. Removed Punctuations,stopwords , 2 length words
10. Lemmatized text

Approach:

Written a method to perform data splitting
Generated ClassFrequencies and Inverse-Class Frequency
Generated vocabulary for k = 600
Trained a naive Bayes model for 80:20 train test data , and obtained accuracies :

Train : `0.9249099473538376`
Test : `0.7973306969846762`

Confusion matrix :

```
array([[297,  20,  47,  56,  43],
       [ 13, 357,  34,  28,  15],
       [ 10,  20, 394,  12,  16],
       [  4,   9,  14, 204,   6],
       [ 11,  18,  21,  13, 361]])
```

Further trained model for 50:50 data split and obtained accuracies :

Train : `0.9361702127659575`
Test : `0.8002203452074917`

Confusion matrix :

```
array([[417,  30,  35,  99,  35],
       [ 22, 473,  35,  43,  33],
       [ 16,  19, 529,  31,  14],
       [  6,  11,   7, 263,   8],
       [ 22,  21,  26,  31, 497]])
```

Further trained model for 70:30 data split and obtained accuracies :

Train :0.9357188093730209
Test :0.8080536912751678

Confusion matrix:

```
array([[340,  25,  35,  69,  33],
       [ 16, 389,  27,  43,  29],
       [  9,  14, 417,  27,  10],
       [  6,   8,  10, 222,   6],
       [  9,  17,  12,  24, 438]])
```