README-

Q1) Scoring and Term-Weighting

Jaccard Coefficient-

Methodology-

i) Preprocessed the given dataset
ii)returned tokens list of individual files as set after preprocessing
iii) preprocessed the query, into tokens as set
iv)Used & for intersection of the sets,(filenumber1 & query) and so on.
v) Used | for union of the sets,(filenumber1 | query) and so on.
vi) using formula to calculate the jaccard coefficient, calculated it and stored it in a dictionary, for individual files.
vii) Used Counter for retrieving the topmost 5 documents.

Preprocessing-

1) Changed all the words to their lower cases
2) Used TweetTokenizer() for tokenizing the string into words
3) Using stopwords, remove all the stop words such as not and in from the token list.
4) Removed all the punctuations from the words in the token list
5) Removed all the blank spaces from the token list

Tf-Idf Matrix

Methodology-

i) Preprocessed the given dataset
ii)returned tokens list of individual files as set after preprocessing
iii) preprocessed the query, into tokens as set
iv) calculated the IDF value of each word is calculated using the formula as mentioned below:

Using smoothing:-
IDF(word)=log(total no. of documents/document frequency(word)+1)
v) Used all 5 weighting schemes for term frequency calculation and report the TF-IDF score and results for each scheme separately.
vi) Computed tf-idf for each doc-val pair and stored it.
vii) Calculated tf-idf score for the query using the matrix

Preprocessing-

1) Changed all the words to their lower cases
2) Used TweetTokenizer() for tokenizing the string into words
3) Using stopwords, remove all the stop words such as not and in from the token list.
4) Removed all the punctuations from the words in the token list
5) Removed all the blank spaces from the token list.

State the pros and cons of using each scoring scheme to find the relevance of documents in your report.

Jaccard Coefficient-

Pros-

i) We can use both categorical as well as continuous values in case of jaccard coefficient.
ii) For word duplication, we should use the jaccard coefficient as duplication does not matter in jaccard coefficient.

Cons-

i)Jaccard coefficient does not work well or accurately with nominal dataset.It means that there is no ranking in the dataset.
ii) There is no term frequency accountable in the jaccard coefficient.
iii)Jaccard coefficient does not follow that if a term occurs rarely in the data, it would be more factual or informative than a term which occurs frequently.

TF-IDF

Pros-

i)It measures accurately the uniqueness as well as the relevance of the contents in the document.
ii)We can associate relevance to each word in the document.
iii)It's easily computable and understandable.

Cons-
i) It does not take into account the semantics of the words.
ii) The computation of the similarity between documents is directed in word count, which is slow for large numbers of words.
iii) It does not consider semantics, the positions of the words in the text, the co-occurrences in different docs, following the Bag of Words model.

Q2) Ranked-Information Retrieval and Evaluation

Methodology-

1) **Consider the queries with qid:4, and store them.**
2) **Calculated the max DCG for sorted as well as unsorted relevance score.**
3) **Made a file rearranging the query-url pairs in order of max DCG.**
4) **Computed nDCG for 50 documents, and the whole set**
5) **Using only feature 75(sum of TF-IDF on the whole document) i.e. the higher the value, the more relevant the URL. Plot a Precision-Recall curve for query "qid:4".**

Q3) Naive Bayes Classifier

Methodology-

1) Firstly, we import the data and perform preprocessing.
2) After this we form the data by appending the target values.
3) Then we split the data into a train and a test set.
4) Then we find tf, cf-icf for all the words of the classes.
5) After this we sort the words according to their tf-icf values and pick words with top k values.
6) Now apply naive bayes using the above k selected features.

Preprocessing-

1) Removing headers and footers
2) Lemmatization on the tokens
3) Converting numbers to string
4) Removing apostrophe
5) Using stopwords, remove all the stop words such as not and in from the token list.
6) Removed all the punctuations from the words in the token list
7) Changed all the words to their lower cases
8) Removed all the blank spaces from the token list

Accuracies achieved are as followed:
For 80:20 train - test ratio : 0.892
For 70:30 train - test ratio : 0.876
For 50:50 train - test ratio : 0.876

Analysis-

These values show that as we decrease the data to train our model, the accuracy decreases, which is absolutely justified because as we increase data for training we get a bit more diverse data and less generalized decreasing the errors because of that.