CSE508: Information Retrieval
Assignment 2

Max Marks: 100

Instructions-

- The assignment is to be attempted in groups (max 2 members).
- Language allowed: Python
- For plagiarism, institute policy will be followed.
- You need to submit README.pdf and code files. The code should be well commented.
- You are allowed to use libraries such as NLTK for data preprocessing.
- Mention methodology, preprocessing steps, and assumptions you may have in README.pdf.
- You will be required to use Github for code management.
  - Each group will create a GitHub repository with the name IR2022_A2_GroupNo (Eg - IR2022_A2_1 for Group No-1).
  - Each group would add the assigned TA as a collaborator to the GitHub repository. TAs' GitHub handles would be shared shortly.
  - While uploading on Classroom, each group would need to upload a link of the GitHub repository. Only one member needs to submit.
- You cannot use any exact API/library for the tasks you have been assigned. You have to do the implementation from scratch. For instance, if you have been asked to implement IDF then you can't use API for the same.
- You will have 10 days to complete the assignment.

Question 1 - [40 Points] Scoring and Term-Weighting

Jaccard Coefficient [20 points]

The goal is to find the Jaccard coefficient between a given query and the document. The formula used is mentioned below as:

Jaccard Coefficient = Intersection of (doc,query) / Union of (doc,query)

The high the value of the Jaccard coefficient, the more the document is relevant for the query.

1. Use the same data given in assignment 1 and carry out the same preprocessing steps as mentioned before.
2. To calculate this make set of the document token and query token and perform intersection and union between the query and each document.
3. Report the top 5 relevant documents based on the value of the Jaccard coefficient.

TF-IDF Matrix [20 points]

The goal is to generate a TF-IDF matrix for each word in the vocab and obtain a TF-IDF score for a given query. TF-IDF has two parts Term Frequency and Inverse Document Frequency.

- Computing Term Frequency involves calculating the raw count of the word in each document and stored as a nested dictionary for each document.
- To calculate the document frequency of each word, find the postings list of each word and subsequently find the no. of documents in each posting list of each word.
- The IDF value of each word is calculated using the formula as mention below:

  Using smoothing:-

  IDF(word)=log(total no. of documents/document frequency(word)+1)
- The Term Frequency is calculated using 5 different variants:

Weighting Scheme	TF Weight
Binary	0,1
Raw count	f(t,d)
Term frequency	f(t,d)/$\sum f(t', d)$
Log normalization	log(1+f(t,d))
Double normalization	0.5+0.5*(f(t,d)/ max(f(t',d))

1. Use the same data given in assignment 1 and carry out the same preprocessing steps as mentioned before.
2. Build the matrix of size no. of document x vocab size.
3. Fill the tf_idf values in the matrix of each word of the vocab.
4. Make the query vector of size vocab
5. Compute the TF-IDF score for the query using the TF-IDF matrix. Report the top 5 relevant documents based on the score.
6. Use all 5 weighting schemes for term frequency calculation and report the TF-IDF score and results for each scheme separately

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

Question 2 - [25 points] Ranked-Information Retrieval and Evaluation

Use the data file provided here. This has been taken from Microsoft learning to rank dataset, which can be found here. Read about the dataset carefully, and what all it contains.

1. Consider only the queries with qid:4 and the relevance judgement labels as relevance score.
2. (10 points) Make a file rearranging the query-url pairs in order of max DCG. State how many such files could be made.
3. (5 points) Compute nDCG
   (a) At 50
   (b) For the whole dataset
4. (10 points) Assume a model that simply ranks URLs on the basis of the value of feature 75 (sum of TF-IDF on the whole document) i.e. the higher the value, the more relevant the URL. Assume any non zero relevance judgment value to be relevant. Plot a Precision-Recall curve for query "qid:4".

Question 3 - [35 points] Naive Bayes Classifier

Download the 20_newsgroup. You need to pick documents of comp.graphics, sci.med, talk.politics.misc, rec.sport.hockey, sci.space [5 classes] for text classification.

Implement the **Naive Bayes algorithm** for text classification using **TF-ICF**, (a modification of **TF-IDF**) as a feature selection technique.

TF-ICF score for a given term belonging to a class can be calculated as follows:

Term Frequency (TF): Number of occurrences of a term in all documents of a particular class
Class Frequency (CF): Number of classes in which that term occurs

Inverse-Class Frequency (ICF): log(N / CF), where N represents the number of classes

Implementation Points:

1. Perform suitable pre-processing steps for the given dataset.
2. Split your dataset randomly into train: test ratio. You need to select the documents randomly for splitting. You are not supposed to split documents in sequential order, for instance, choosing the first 800 documents in the train set and last 200 in the test set for the train: test ratio of 80:20.
3. Implement the TF-ICF scoring technique for efficient feature selection. Select the top k features for each class. Subsequently, the effective vocabulary shall be the union of the top k features of each class.
4. For each class, train your Naive Bayes Classifier on the training data.
5. Test your classifier on testing data and report the confusion matrix and overall accuracy.
6. Perform the above steps on 50:50, 70:30, and 80:20 training and testing split ratios.
7. Analyze the performance of the classification algorithm for the feature selection technique across different train: test ratios.